Fakultät Informatik
Professur Rechnernetze

Konzeption und prototypische Realisierung einer Embedded Linux Distribution für sichere WLAN Access Points

Diplomarbeit

Daniel Woithe

Dresden, den 11. August 2005

Technische Universität Dresden/Fakultät Informatik

Aufgabenstellung für die Diplomarbeit

Name, Vorname: **Woithe, Daniel**

Studiengang: Informatik Matr. Nr.|2||7||8||1||7||9||5|

Thema:

Konzeption und prototypische Realisierung einer Embedded Linux Distribution für sichere WLAN Access Points

Zielstellung :

Drahtlose Netze wie WLAN nach 802.11 erfreuen sich einer hohen Beliebtheit, da sie einen flexibleren Netzzugang ermöglichen. Andererseits ist die Kommunikation wesentlich höheren Risiken ausgesetzt. Dafür wurden spezielle Sicherheitsprotokolle entwickelt, die allerdings zum Teil Schwachstellen aufweisen. Da den bisher verfügbaren Sicherheitsmechanismen nur unzureichend vertraut wird, ist ein breiter Einsatz in Unternehmensumgebungen derzeit kaum Realität.
Ziel der Arbeit ist die Konzeption einer Linux Distribution, die auch hohen Sicherheitsanforderungen gerecht wird. Dazu sind zunächst relevante Technologien vorzustellen. Im Anschluss sind existierende WLAN Distributionen und Router zu analysieren. Darauf basierend ist eine Konzeption für einen Enterprise WLAN Access Point zu erstellen. Den Schwerpunkt der Arbeit bildet die praktische Umsetzung auf ein INTEL Netzwerkprozessorsystem. Dabei sind folgende Funktionen zu berücksichtigen: WLAN Funktionalität, Firewall, VPN, Authentifizierung, Accounting sowie die Unterstützung mehrerer Benutzergruppen. Den Abschluss der Arbeit bilden eine Bewertung der Lösung im Vergleich mit herkömmlichen Systemen sowie ein Ausblick zu offenen Problemen.

Schwerpunkte:

- Übersicht WLAN Technologie und Sicherheit
- Evaluation von WLAN Distributionen und Routern
- Vorstellung der Hardwareplattform (INTEL IXP425 Netzwerkprozessor)
- Konzeption für einen Enterprise WLAN Access Point
- Praktische Realisierung und Test
- Bewertung der Lösung

Betreuer: Dipl.-Inform. Mirko Benz, Dipl.-Inform. Stephan Groß

Verantwortlicher Hochschullehrer: Prof. Dr. habil. Alexander Schill

Institut: Systemarchitektur

Beginn am : 15.02.2005

Einzureichen am : 15.08.2005

_____

Unterschrift des verantwortlichen Hochschullehrers

Verteiler: 1 x Prüfungsamt, 1x HSL, 1x Betreuer, 1x Student

Inhaltsverzeichnis

1 Einleitung **8**
- 1.1 Motivation . . . 8
- 1.2 Ziel der Arbeit . . . 9
- 1.3 Gliederung und Inhalt . . . 9

2 Grundlagen der Sicherheit von Wireless LAN **11**
- 2.1 Wireless LAN Technologie . . . 11
- 2.2 Schutzziele . . . 13
- 2.3 Sicherheit in IEEE 802.11 . . . 13
- 2.4 WEP-Sicherheitsarchitektur . . . 14
  - 2.4.1 Funktionsweise . . . 14
  - 2.4.2 Schwachstellen der WEP-Architektur . . . 16
- 2.5 WPA-Sicherheitsarchitektur . . . 18
- 2.6 Der IEEE 802.11i Standard . . . 21
- 2.7 Das Authentifizierungsprotokoll IEEE 802.1x . . . 22
  - 2.7.1 Grundlagen von IEEE 802.1x . . . 22
  - 2.7.2 Authentifizierung mit IEEE 802.1x . . . 23
- 2.8 Das Extensible Authentication Protokoll (EAP) . . . 25
  - 2.8.1 Aufbau eines EAP-Paketes . . . 25
  - 2.8.2 Beispiel zum EAP-Austausch . . . 26
  - 2.8.3 EAP over LAN (EAPOL) . . . 26
  - 2.8.4 EAP-TLS . . . 27
  - 2.8.5 EAP-TTLS . . . 28
  - 2.8.6 PEAP . . . 28
  - 2.8.7 Zusammenfassung und Vergleich der EAP-Methoden . . . 29
- 2.9 Sicherung von WLANs durch IPSec . . . 29
  - 2.9.1 Transport Mode und Tunnel Mode . . . 30
  - 2.9.2 Authentication Header . . . 31
  - 2.9.3 Encapsulation Security Payload . . . 32
  - 2.9.4 Internet Key Exchange . . . 33
  - 2.9.5 Fazit . . . 35

3 Die Intel-Hardwareplattform **36**
- 3.1 Der Intel IXP425 Netzwerkprozessor . . . 36
- 3.2 Die Intel Netzwerk-Gateway-Referenzplattform IXDPG425 . . . 37

4 Die Entwicklungsumgebung OpenEmbedded 39
4.1 Aufbau von OpenEmbedded . . . 39
4.2 Konfiguration von OpenEmbedded . . . 40

5 Konzeption einer Linux Distribution für einen WLAN Access Point 43
5.1 Analyse kommerzieller WLAN Access Points . . . 43
5.1.1 Die Cisco-Aironet Serie . . . 43
5.1.2 Der Bintec X2250 . . . 45
5.1.3 Zusammenfassung . . . 46
5.2 Einsatzszenarien für WLAN Access Points . . . 46
5.2.1 WLAN Access Point ohne Sicherheitsfunktionen . . . 47
5.2.2 WLAN Access Point mit IPSec-Sicherheit . . . 47
5.2.3 WLAN Access Point mit IEEE 802.11i . . . 48
5.2.4 WLAN Router . . . 49
5.2.5 WLAN Router mit unterschiedlichen Benutzergruppen . . . 51
5.3 Realisierung eines WLAN Access Points mit Open Source Software . . . 52
5.3.1 Erstellung einer Toolchain . . . 53
5.3.2 Auswahl eines Kernels und einer Kernelkonfiguration . . . 54
5.3.3 Die Intel Access Library . . . 55
5.3.4 WLAN-Treiber und Wireless-Tools . . . 56
5.3.5 Unterstützung kryptographischer Operationen . . . 57
5.3.6 IPSec für Virtuelle Private Netzwerke . . . 59
5.3.7 Unterstützung von IEEE 802.11i (WPA, WPA2) . . . 60
5.3.8 Firewall . . . 61
5.3.9 Zusätzliche Netzwerkdienste . . . 62
5.3.10 Netzwerktest und -diagnose . . . 63

6 Praktische Umsetzung der WLAN Distribution für den IXDPG425 64
6.1 Kompilierung der Module . . . 64
6.1.1 Kompilieren des Madwifi-Treibers . . . 65
6.1.2 Integration der Krypto-Unterstützung in die Intel Access Library . . . 66
6.1.3 Integration des OCF in den Linux-Kernel . . . 66
6.1.4 Kompilierung und Test von Dnsmasq . . . 66
6.1.5 Erweiterung zur Unterstützung mehrerer Benutzergruppen . . . 67
6.2 Konfiguration und Test des WLAN Access Points . . . 69
6.2.1 Erstellen von X.509-Zertifikaten . . . 70
6.2.2 Konfiguration des WLAN Adapters . . . 73
6.2.3 Konfiguration des Hostap-Dämon . . . 73
6.2.4 Test der WLAN Verbindung . . . 74
6.2.5 Konfiguration und Test von IPSec . . . 75
6.2.6 Konfiguration der Firewall . . . 77
6.2.7 Geschwindigkeitsanalyse . . . 77
6.3 Konfiguration der Clients . . . 79
6.3.1 IPSec mit Linux . . . 79
6.3.2 Natives IPSec mit Windows 2000/XP . . . 79
6.3.3 IPSec über L2TP . . . 79

7 Validierung **81**
7.1 Testumgebung 81
7.2 WLAN Access Point ohne Sicherheitsfunktionen 82
7.3 WLAN Access Point mit IPSec-Sicherheit 83
7.4 WLAN Access Point mit IEEE 802.11i 83
7.5 WLAN Router 84
7.6 WLAN Router mit unterschiedlichen Benutzergruppen 85

8 Fazit und Perspektiven **87**
8.1 Zusammenfassung 87
8.2 Mögliche Erweiterungen 88
8.3 Bewertung der Lösung 89

A Konfigurationsdateien **91**
A.1 Dnsmasq 91
A.2 IPSec 92
A.3 L2TP 94
A.4 PPP 96
A.5 Hostap-Dämon 96
A.6 Shorewall 100

B Inhalt der beiliegenden DVD **103**

C Abkürzungsverzeichnis **104**

D Literaturverzeichnis **107**

Abbildungsverzeichnis

2.1 WEP-Verschlüsselung 15
2.2 WEP-Integritätssicherung 15
2.3 WEP-Fehlerkorrektur 16
2.4 WEP-Authentifizierung 16
2.5 IEEE 802.1x Port Based Access Control 20
2.6 Supplicant, Authenticator und Authentication Server im Protokoll IEEE 802.1x 24
2.7 Beispiel EAP-Austausch 26
2.8 IP-Datagramm im Transport Mode 30
2.9 IP-Datagramm in Tunnel Mode 30
2.10 IP-Datagramm mit AH im Transport Mode 31
2.11 Datagramm mit ESP im Transport Mode 32

3.1 Aufbau des IXP425 37
3.2 Aufbau des IXDPG425 38

5.1 WLAN Access Point ohne Sicherheitsfunktionen 47
5.2 WLAN Access Point mit IPSec 48
5.3 WLAN Access Point mit WPA-Enterprise 49
5.4 WLAN Router 50
5.5 WLAN Router mit Multi-SSID 51
5.6 Asynchrone Verschlüsselung mit OCF 58

6.1 Ergebnisse Geschwindigkeitsmessungen IXDPG425 78

7.1 Testumgebung 82

Tabellenverzeichnis

2.1 Aufbau EAP-Paket 25
2.2 EAP-Codes 26
2.3 EAPOL-Typen 27
2.4 EAP-Methoden im Vergleich 29

4.1 Tasks in OpenEmbedded 40

6.1 Module der WLAN Distribution 65
6.2 Übersicht Konfigurationsdateien 69

B.1 Inhalt der beiliegenden DVD 103

Kapitel 1

Einleitung

1.1 Motivation

Seit der Verabschiedung des Standards IEEE 802.11 im Jahre 1997 kann die drahtlose Vernetzung mit Wireless LAN stets steigende Marktanteile verzeichnen. Während noch vor einigen Jahren die Errichtung von Funkverbindungen mit dem Erwerb von teurer, nicht standardisierter Spezialhardware verbunden war und daher häufig nur in Unternehmensumgebungen für Punkt-zu-Punkt-Verbindungen genutzt wurde, sind heute WLAN-Komponenten für jedermann erschwinglich. So sind in den USA bereits heute mehr als die Hälfte der Heimnetzwerke mit der Wireless LAN (WLAN)-Technologie ausgestattet und nahezu jedes auf dem Markt befindliche Notebook besitzt die Fähigkeit, mit einem WLAN-Access-Point zu kommunizieren.

Diese starke Akzeptanz der WLAN-Technologie ist bisher jedoch nur im Segment der Privatanwender zu verzeichnen. In großen Unternehmen wird, nicht zuletzt wegen der häufigen Medienberichte über unzureichend geschützte Access-Points, der Erweiterung der traditionellen drahtgebundenen Vernetzung durch drahtlose Lösungen mit Misstrauen begegnet. Diese Vorsicht scheint angesichts der schwachen Sicherheitseigenschaften gängiger 802.11-Produkte auch angemessen zu sein, vor allem dann, wenn die Funkverbindung außerhalb des Firmengebäudes bzw. der Grundstücksgrenzen wahrnehmbar ist.

Aus diesem Grund werden zur Etablierung von drahtlosen Netzen in Unternehmensumgebungen Sicherheitsfunktionen benötigt, die den betrieblichen Anforderungen an Vertraulichkeit, Integrität und Authentizität gerecht werden. Das Institute of Electrical and Electronics Engineers (IEEE) als Standardisierungsgremium sowie die Wireless Fidelity-Allianz (WiFi), eine Herstellervereinigung zur Verbreitung der IEEE-Standards, haben auf diese Sicherheitsbedenken bereits reagiert und veröffentlichten neue Sicherheitsarchitekturen, welche die bisherige Wired Equivalent Privacy (WEP)-Sicherheitsfunktionen ablösen sollen. Während mit WiFi Protected Access (WPA) eine Interimslösung zur stärkeren Absicherung von Funknetzen geschaffen wurde, liegt nun mit dem am 25. Juni 2004 ratifizierten IEEE-Standard 802.11i eine Sicherheitsarchitektur vor, die unter der Verwendung des Advanced Encryption Standard (AES) eine ausreichende Sicherheit garantieren soll.

Die Hersteller von WLAN Access Points haben die Notwendigkeit robusterer Sicherheitsarchitekturen erkannt, und ihre Hard- und Software um die in IEEE 802.11i definierten

Verfahren erweitert. Hierbei werden jedoch häufig nur Teilmengen sowie herstellerabhängige Erweiterungen dieser Norm implementiert.

Doch auch im Umfeld der Open Source Softwareentwicklung existiert eine Vielzahl von Projekten, die sich mit der Absicherung von drahtlosen Netzwerkverbindungen beschäftigen. Dabei werden neben den in IEEE 802.11i vorgestellten Ideen auch alternative Sicherheitskonzepte umgesetzt.

Die vorliegende Arbeit bewertet die Eignung der existierenden Sicherheitsarchitekturen im Unternehmenskontext und stellt ein Konzept für einen auf Open Source Software basierenden Enterprise WLAN Access Point vor.

1.2 Ziel der Arbeit

In dieser Diplomarbeit soll eine Embedded Linux Distribution für einen Enterprise WLAN Access Point realisiert werden, der die flexible und erweiterbare Nutzung aktueller Sicherheitsarchitekturen ermöglicht. Als Zielplattform soll dabei ein auf dem Netzwerkprozessor Intel IXP425 basierendes Netzwerkgateway IXDPG425 eingesetzt werden.

Die zu realisierende Embedded Linux Distribution wird zur Erbringung der gewünschten Funktionalität aus einer Vielzahl von unterschiedlichen Open Source Softwarekomponenten zusammengestellt. Jede einzelne Komponente muss dabei auf die Zielplattform angepasst, kompiliert, konfiguriert sowie getestet werden. Außerdem ist die Interaktion der verschiedenen Softwaremodule innerhalb der Embedded Linux Distribution sicherzustellen und zu testen.

Können spezielle Anforderungen an den Access Point nicht durch existierende Projekte realisiert werden, so sind alternative Lösungswege, beispielsweise durch eigene Programme oder Programmanpassungen, darzustellen.

1.3 Gliederung und Inhalt

Für die Realisierung der Embedded Linux Distribution werden unterschiedliche Sicherheitskonzepte mit einer Vielzahl von Protokollen und Verfahren eingesetzt. Die im Kontext der WLAN Sicherheit bedeutendsten Verfahren sollen im Kapitel 2 vorgestellt und auf ihre Eignung im Unternehmensbereich überprüft werden.

Im darauf folgenden Kapitel 3 erfolgt eine kurze Beschreibung der Intel Hardwareplattform, für welche die WLAN Distribution realisiert werden soll. Die zur Realisierung eingesetzte Entwicklungsumgebung „OpenEmbedded“ wird im 4. Kapitel vorgestellt. Außerdem soll dabei die Anpassung dieser Software an die Erfordernisse der Hardwareplattform dargestellt werden.

Als Grundlage für die spätere Realisierung der Embedded Linux Distribution sollen im Kapitel 5 Konzepte für einen Enterprise WLAN Access Point vorgestellt werden. Dafür

wurden zunächst existierende WLAN Router analysiert und darauf aufbauend unterschiedliche Szenarien für einen Wireless LAN Access Point zusammengestellt. Die Zuordnung der in den Szenarien erforderlichen Funktionalitäten zu Open Source Softwarekomponenten ist ebenfalls Teil dieses Abschnitts.

Die praktische Umsetzung der Konzeption, die auch den Schwerpunkt dieser Arbeit und damit den deutlich höchsten Zeitaufwand darstellt, soll im Kapitel 6 exemplarisch präsentiert werden. Aufgrund der Vielzahl der genutzten Softwarekomponenten und dem sich daraus ergebenen Aufwand für Konfiguration und Test, werden in diesem Abschnitt nur spezielle Beispiele der Realisationsphase vorgestellt. Ein Integrationstest der in der Konzeptionsphase entstandenen Szenarien erfolgt im Kapitel 7.

Die Ergebnisse dieser Diplomarbeit werden im Kapitel 8 zusammengefasst und diskutiert. Des Weiteren soll hierbei ein Ausblick auf zukünfigte Erweiterungen und Fortsetzungen dieser Arbeit gegeben werden.

Kapitel 2

Grundlagen der Sicherheit von Wireless LAN

Der Begriff Wireless LAN (WLAN) bezeichnet ein „drahtloses“ lokales Funknetzwerk. Nahezu alle derzeit erhältlichen WLAN-Produkte basieren im wesentlichen immer noch auf dem im Jahre 1997 veröffentlichten Standard der IEEE mit der Bezeichnung 802.11. Zur Sicherung der Funkübertragung gegenüber unberechtigten Dritten wurde schon damals die WEP-Sicherheitsarchitektur in diesen Standard integriert, welche sich jedoch in den letzten Jahren als völlig unzureichend erwiesen hat.

In diesem Abschnitt sollen nun die WLAN-Technologie, die WEP-Sicherheitsarchitektur mitsamt ihrer Schwachstellen sowie Erweiterungen bezüglich der Sicherheit von Funknetzen vorgestellt werden.

2.1 Wireless LAN Technologie

Funkverbindungen zwischen Computern existieren bereits seit Anfang der 90er Jahre, jedoch konnten sich diese proprietären Produkte aufgrund geringer Übertragungsleistungen und fehlender Kompatibilität nie einen großen Markt erschließen. Erst mit der im Jahre 1997 abgeschlossenen Standardisierung einer drahtlosen Netzwerkverbindung, veröffentlicht unter der Bezeichnung IEEE 802.11, wurde der Grundstein für die heute weit verbreitete WLAN-Technologie gelegt. Der Standard 802.11 beschreibt drei verschiedene Techniken zur drahtlosen Übertragung: zwei funkbasierte im Frequenzbereich von 2,4 bis 2,5 GHz, mit den unterschiedlichen Bandspreizverfahren Frequency Hopping Spread Spectrum (FHSS) und Direct Sequence Spread Spectrum (DSSS), und eine auf Infrarot basierende.

Während anfänglich nur Datenübertragungsraten von bis zu 2 MBit/s möglich waren, konnte mit diversen Erweiterungen eine deutliche Zunahme der Geschwindigkeit erreicht werden.

Die ersten Erweiterungen aus dem Jahre 1999 tragen die Bezeichnung 802.11a und 802.11b. Der Standard 802.11a beschreibt eine physikalische Schicht zur Datenübertragung, welche im 5 GHz-Bereich Übertragungsgeschwindigkeiten von bis zu 54 MBit/s erreichen kann. Aufgrund des Frequenzbereiches konnten diese Geräte jedoch anfänglich nur in den USA

eingesetzt werden, seit Ende 2002 ist die Benutzung auch in Deutschland erlaubt, jedoch mit deutlich verminderten Sendeleistungen.

Durch eine Optimierung der Übertragungstechnologie konnte unter Wahrung der Kompatibilität mit der 802.11-DSSS-Technik im 2,4GHz-Bereich mit dem Standard 802.11b im Jahr 1999 eine Steigerung der Übertragungsleistung auf bis zu 11 MBit/s erreicht werden. Vor allem die im selben Jahr gegründete Herstellervereinigung WiFi-Alliance, die Produkte auf ihre Kompatibilität zum 802.11b-Standard prüft und das WiFi-Logo verleiht, konnte sich mit dieser Technologie einen breiten Markt erschließen.

Eine abermalige Leistungssteigerung ermöglicht die im Jahre 2003 veröffentlichte Norm IEEE 802.11g, die Datenübertragungen mit bis zu 54 MBit/s Bruttodatenrate im 2,4 GHz Industrial, Scientific and Medical Band (ISM-Band) ermöglicht. Hierbei ist auch die Abwärtskompatibilität zu Geräten der 802.11b-Serie gewährleistet - ein entscheidender Vorteil gegenüber den im 5 GHz-Bereich operierenden Geräten des 802.11a-Standards.

Bei allen auf IEEE 802.11 sowie der drei beschriebenen Erweiterungen basierenden Wireless LAN-Geräte sind zwei unterschiedliche Betriebsmodi möglich: Ad-Hoc- sowie Infrastruktur-Modus.

Im Ad-Hoc-Modus kommunizieren verschiedene Rechner im Punkt-zu-Punkt-Betrieb direkt miteinander. Ein Punkt-zu-Punkt-Netz kann zwei oder mehrere Rechner umfassen, die mit jeweils einem WLAN-Adapter ausgerüstet sind. Alle teilnehmenden Adapter müssen dabei auf denselben Kanal eingestellt sein und können bei entsprechender räumlicher Nähe direkt Daten miteinander austauschen. Außer den WLAN-Adaptern werden keine weiteren Geräte benötigt, weshalb dieser Modus die kostengünstigste Variante für eine drahtlose Vernetzung darstellt. Jedoch kommen in diesem Betriebsmodus meist nur ungenügende Sicherheitsmechanismen zur Anwendung - ein Angreifer kann durch die Konfiguration des richtigen Kanals sofort Zugriff zum Ad-hoc-Netzwerk erlangen.

Bei Verwendung eines zentralen Zugriffspunktes, eines sogenannten Access Point (AP), arbeitet das drahtlose Netz im Infrastruktur-Modus. Hierbei wird durch den AP eine Funkzelle mit einer Größe von 30 bis 300 Metern etabliert, in welcher die Clients über ihn kommunizieren können. Häufig dient der AP auch als Schnittstelle zum kabelgebundenen LAN. Eine Vergrößerung des Funkbereiches kann durch den Einsatz mehrerer Access Points mit überlappenden Funkzellen erreicht werden.

Während im Ad-hoc-Modus keine zentrale Instanz zur Festlegung von Sicherheitsparametern existiert, kann diese Aufgabe im Infrastruktur-Modus vom Access Point übernommen werden. Durch den AP werden hierbei die Regeln zur Authentifizierung und Verschlüsselung festgelegt, d.h. Clients können nur bei erfolgreicher Anwendung dieser Sicherheitsregeln das Netzwerk nutzen.

Der genaue Aufbau dieser Sicherheitsmechanismen sowie mögliche Angriffe sollen in den folgenden Kapiteln vorgestellt werden. Zuvor folgt jedoch eine Darstellung der Schutzziele in Bezug auf die Absicherung von WLAN-Systemen.

2.2 Schutzziele

Bei der Absicherung von Wireless-LAN-Verbindungen wird mit Hilfe von kryptographischen Methoden und anderen Techniken versucht, die Schutzziele Vertraulichkeit, Integrität, Authentizität und Verfügbarkeit mit jeweils unterschiedlichen Gewichtungen zu erreichen. Diese Schutzziele sollen nun im Detail erläutert werden.

Vertraulichkeit Das zumeist naheliegendste Ziel beim Schutz von Daten ist die Vertraulichkeit, also die Geheimhaltung von Informationen vor allen nicht zum Zugriff berechtigten Personen. Vertraulichkeit wird durch die Anwendung kryptographischer Verfahren erreicht. Im Kontext der drahtlosen Verbindungen bedeutet dies, dass sowohl Nutzdaten als auch Verbindungsdaten, die zum Aufbau, zur Aufrecherhaltung und zum Abbau einer Datenübertragung benötigt werden, nicht von unberechtigten Dritten entschlüsselt werden können, selbst wenn das Empfangen der entsprechenden Datenpakete gelingt.

Integrität Die Integrität stellt den Schutz der Daten vor ungewollter Veränderung dar. Dies bedeutet für den Empfänger im WLAN, dass er sich sicher sein kann, dass die von ihm empfangenen Daten auch genau so vom Sender abgeschickt wurden. Im Falle einer Manipulation der Daten auf dem Verbindungskanal muss der Empfänger diese sicher erkennen können. Der Integritätssicherung ähnlich ist der Schutz gegen zufällige Übertragungsfehler beim Übermitteln der Nachricht. Dieser kann Bitfehler entdecken, aber im Allgemeinen nicht die Authentizität von Daten garantieren.

Authentizität Um eine Zugangskontrolle zu ermöglichen, muss vor dem Bereitstellen von Daten oder Diensten die Authentizität der Kommunikationspartner sichergestellt werden. Die Authentizität, die eng mit der Integrität verbunden ist, stellt sicher, dass neben der unveränderten Übertragung auch die Herkunft und das Ziel der Daten zweifelsfrei bekannt sind. Damit wird die Grundlage für Verbindlichkeit geschaffen.

Verfügbarkeit Verfügbarkeit bedeutet, dass ein Dienst immer dann zur Verfügung steht, wenn ein berechtigter Nutzer diesen in Anspruch nehmen möchte. Beeinträchtigt werden kann die Verfügbarkeit beispielsweise durch Überlastung oder Sabotage. Ziel ist es, Störungen bei der Dienstbereitstellung auszuschließen bzw. zumindest soweit wie möglich einzugrenzen. Im Fall einer Störung sollte diese sicher erkannt werden und Maßnahmen eingeleitet werden, die die Verfügbarkeit wiederherstellen können.

2.3 Sicherheit in IEEE 802.11

Sowohl in den verschiedenen Standards der IEEE-Gruppe 802.11 als auch in den auf dem Markt befindlichen Access Points existieren verschiedene Sicherheitsmaßnahmen, um die im letzten Abschnitt beschriebenen Schutzziele zu erreichen.

Als ein einfacher Mechanismus zur Zugriffskontrolle in WLAN-Netzen dient die Media Access Control (MAC)-Adresse, eine Hardwarekennung, mit welcher bereits vom Hersteller alle WLAN-Adapter gekennzeichnet werden. Ein im Access Point implementierter

Adressfilter ist in der Lage, nur die in einer Liste eingetragenen MAC-Adressen zur Benutzung der Infrastruktur zuzulassen. Leider besteht bei vielen Karten die Möglichkeit, die MAC-Adresse über eine Software zu ändern. Der Filter bietet daher nur einen zusätzlichen Schutz, ist jedoch keinesfalls ausreichend.

Den eigentlichen Sicherheitsmechanismus des 802.11-Standards stellt die WEP-Sicherheitsarchitektur dar, die versucht, die Schutzziele Vertraulichkeit, Integrität und Authentizität unter Verwendung von Kryptographie zur Verschlüsselung und Authentisierung zu erreichen. Diese Methoden sowie die Gründe für den nur unzureichenden Schutz der WEP-Architektur werden im nächsten Abschnitt diskutiert.

2.4 WEP-Sicherheitsarchitektur

Zur Absicherung der Funkverbindung zwischen Client und Access Point wurde die WEP-Sicherheitsarchitektur in die IEEE 802.11-Normierung einbezogen. Die folgenden Sicherheitsziele sollen durch WEP erreicht werden: Vertraulichkeit, Integrität und Authentizität. Für die Vertraulichkeit kommt die Stromchiffre RC4 zum Einsatz, eine CRC - Summe (Cyclic Redundancy Check) soll die Integrität gewährleisten und die Authentifizierung erfolgt über ein Challenge-Response-Verfahren.

2.4.1 Funktionsweise

Die WEP-Architektur nutzt den RC4-Stromchiffrier-Algorithmus zur Verschlüsselung der über das WLAN gesendeten Nachrichten. Für Verschlüsselung und Entschlüsselung kommt dabei ein gemeinsamer Schlüssel mit einer Länge von 40 bzw. 104 Bit zum Einsatz, der sowohl dem Access Point als auch allen beteiligten Clients bekannt sein muss. Gemäß dem Standard 802.11 können in jedem Gerät bis zu 4 verschiedene Schlüssel hinterlegt werden, die Schlüsselzuordnung erfolgt dabei über eine Key-ID im Frame Body.

Verschlüsselung Zur Verschlüsselung eines Datenpaketes wird folgender Algorithmus angewendet: Die Nachricht M wird im ersten Schritt mit einer 32 Bit langen Prüfsumme (CRC-32) versehen, welche als Integrity Check Value (ICV) bezeichnet wird. Diese wird an das eigentliche Datenpaket angehängt und bildet verkettet mit der Nachricht den Klartext (M || ICV). Im zweiten Schritt wird ein 24 Bit langer Initialisierungsvektor (IV) erzeugt, welcher zusammen mit dem geheimen Schlüssel K verkettet wird. Diese Zeichenkette, je nach Schlüssellänge 64 oder 128 Bit lang, bildet die Grundlage für den Schlüsselstrom RC4 (K || IV) des RC4-Algorithmus. Im letzten Schritt wird aus dem Klartext (M || ICV) über eine XOR-Verknüpfung mit dem Schlüsselstrom RC4 (K || IV) der Schlüsseltext C erzeugt. Das zu übertragende, verschlüsselte Datenpaket, auch Frame Body genannt, besteht dann aus dem Initialisierungsvektor, der Key-ID sowie dem durch den RC4-Algorithmus erzeugten Schlüsseltext.

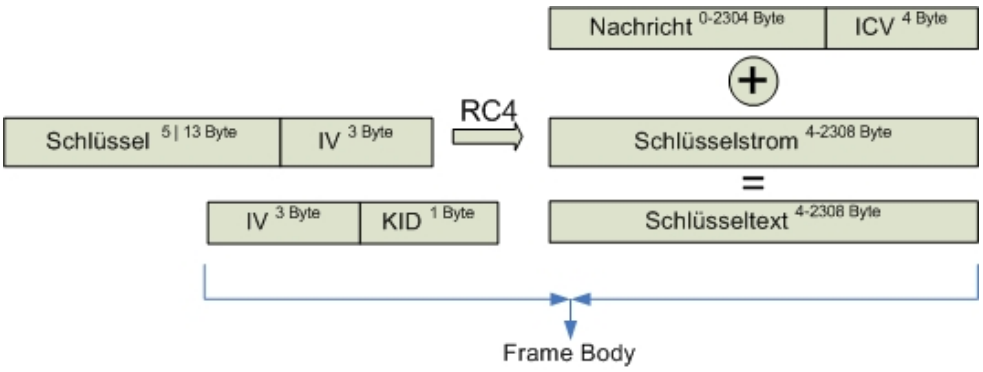

Abbildung 2.1: WEP-Verschlüsselung

Entschlüsselung Mit Hilfe der übertragenen Informationen sowie des geheimen Schlüssels kann nun auf der Empfängerseite die Entschlüsselung durchgeführt werden. Der geheime Schlüssel wird durch die Key-ID bestimmt; der im Klartext übertragene Initialisierungsvektor wird mit diesem verkettet und bildet somit den Schlüsselstrom RC4 (K || IV). Die Entschlüsselung erfolgt analog zur Verschlüsselung, d.h. eine XOR-Verknüpfung des Schlüsseltextes C mit dem Schlüsselstrom RC4 (K || IV) ergibt nun wieder den Klartext (M || ICV). Anschließend findet durch ein erneutes Berechnen der ICV und ein Vergleichen mit dem erhaltenen ICV ein Integritätstest statt.

Integrität Bei der Integritätsüberprüfung kommt eine CRC-32-Prüfsumme zum Einsatz - ein Verfahren, das eigentlich für die Erkennung von zufälligen Fehlern bei der Übertragung von Daten eingesetzt wird. Diese über dem Nachrichtentext M gebildete Prüfsumme ICV wird an die Nachricht angehangen und dadurch auch verschlüsselt übertragen.

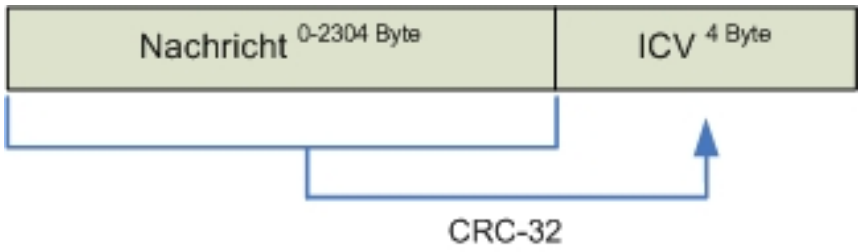

Abbildung 2.2: WEP-Integritätssicherung

Zum Erkennen von Übertragungsfehlern wird wieder das CRC-32 Verfahren angewendet, diesmal jedoch auf den gesamten, zu übermittelnden Frame. Dem nach der Verschlüsselung entstandenen Frame Body wird zuvor ein MAC-Header, der für die Adressierung im 802.11-Netz verantwortlich ist, vorangestellt. Danach wird mit CRC-32 die Frame Check Sequence (FCS) berechnet und an den MAC-Frame angehangen.

Zum Überprüfen einer fehlerfreien Datenübertragung wird nun auf Empfängerseite

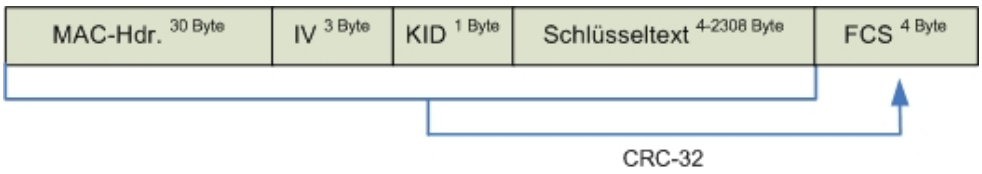

Abbildung 2.3: WEP-Fehlerkorrektur

zuerst ein Vergleich des FCS vorgenommen, bevor in weiteren Schritten die Entschlüsselung und Integritätsüberprüfung durchgeführt wird.

Authentifizierung Der Standard 802.11 erlaubt zwei unterschiedliche Betriebsmodi für die Authentifizierung: Open System und Shared Key. Beim Einsatz der Open System Authentifizierung kann jeder Client sich ohne Durchführung einer Überprüfung am AP anmelden. Im Shared Key Modus wird ein Challenge Response Verfahren eingesetzt, das denselben geheimen Schlüssel wie bei der Verschlüsselung verwendet. Beim Authentifizierungsvorgang muss nun der Client gegenüber dem AP beweisen, dass er den geheimen Schlüssel kennt. Dazu sendet der Client im ersten Schritt eine Authentifizierungsanfrage an den AP, welche seine Identität in Form der MAC-Adresse, eine 16 Bit lange Authentication Algorithm Identification (AAI) zur Auswahl der Authentifizierungsmethode, und eine ebenfalls 16 Bit lange Sequenznummer (SN) enthält. Im zweiten Schritt antwortet der AP mit der gleichen AAI, einer um 1 erhöhten Sequenznummer und einer 128 Byte langen Zufallszahl. Der Client erhält diese Challenge, erhöht die Sequenznummer auf 3, verschlüsselt dieses Paket unter Anwendung der WEP-Verschlüsselung und schickt den Schlüsseltext als Response an den AP. Im letzten Schritt muss nun im AP diese Response überprüft werden, d.h. sie wird entschlüsselt und die erhaltene 128 Byte lange Zufallszahl wird mit der im Schritt 2 versendeten Zufallszahl verglichen. Fällt dieser Vergleich positiv aus erhält der Client die Meldung „Successful“ und ist somit authentifiziert, im negativen Fall würde die Meldung „Unsuccessful“ übertragen werden.

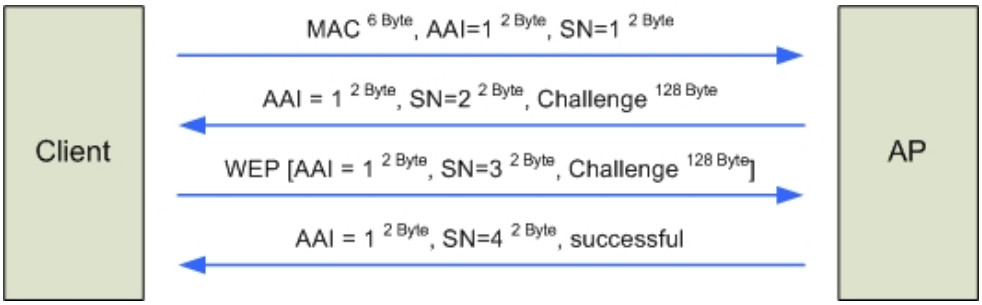
Client
AP
MAC 6 Byte, AAI=1 2 Byte, SN=1 2 Byte
AAI = 1 2 Byte, SN=2 2 Byte, Challenge 128 Byte
WEP [AAI = 1 2 Byte, SN=3 2 Byte, Challenge 128 Byte]
AAI = 1 2 Byte, SN=4 2 Byte, successful

Abbildung 2.4: WEP-Authentifizierung

2.4.2 Schwachstellen der WEP-Architektur

Seit der Veröffentlichung der WEP-Architektur wurde eine Vielzahl von Angriffsmustern auf damit gesicherte WLANs veröffentlicht. Diese Angriffe können in aktive und passive

Angriffe unterschieden werden.

Wenn sich ein Angreifer im Versorgungsbereich eines Funknetzes befindet und Datenpakete mitschneiden und analysieren kann, so findet ein passiver Angriff statt. Im Falle einer aktivierten WEP-Verschlüsselung wird der Angreifer so viel wie möglich verschlüsselte Pakete mit den dazu gehörigen IV's mitschneiden, um diese für einen Angriff auf die Schwachstellen der WEP-Verschlüsselung zu nutzen. Gelingt dies, kann der Angreifer von nun an sämtliche Datenpakete in Echtzeit mitlesen und somit Zugriff auf die interne Netzwerkkommunikation erlangen. Damit wird die gewünschte Vertraulichkeit der Daten nicht mehr gewährleistet. Des Weiteren stellen die durch passive Angriffe erhaltenen Informationen häufig die Grundlage für einen darauf folgenden aktiven Angriff dar. Bei aktiven Angriffen wird die Integrität und Authentizität der Daten durch bewusstes Manipulieren oder Einspielen von Datenpaketen gefährdet.

Im häufigsten Fall ist jedoch gar kein Angriff notwendig, um Zugriff auf die übertragenen Daten zu haben. Im Standard 802.11 wird die WEP-Sicherheitsarchitektur nur als Option angeboten, d.h. fast alle WLAN-Geräte müssen speziell konfiguriert sein, um Authentifikation und Verschlüsselung zu aktivieren - in der Standard-Einstellung sind diese Sicherheitsfunktionen zumeist deaktiviert.

Die größte Angriffsfläche bei WEP-gesicherten Netzwerken stellt die Schlüsselgenerierung und die Schlüsselverteilung dar. Aufgrund der strengen Ausfuhrpolitik kryptographischer Erzeugnisse in den USA konnte zu Beginn der WEP-Entwicklung nur ein 40 Bit Schlüssel eingesetzt werden, welcher mit heutigen Rechnenleistungen innerhalb von wenigen Stunden bis Tagen mit einer Brute-Force-Attacke ermittelt werden kann (Vgl. [Fre03], [Ste04]). Eine Vergrößerung der Schlüssellänge auf 104 Bit schließt zwar diesen Angriff faktisch aus, jedoch kann auch hier bei unvorsichtiger Wahl des Schlüssels mit Hilfe einer Wörterbuchattacke ein Angriff gelingen.

Vor allem in großen Netzwerkumgebungen stellt die sichere Verteilung der gemeinsamen Schlüssel eine kaum zu bewältigende Aufgabe dar. So benötigt jeder Client, der an der Kommunikation teilnehmen will, diesen geheimen Schlüssel, um Zugriff zum Netzwerk zu erlangen. Da der Standard weder Vorgaben zum Wechseln der Schlüssel noch einen Automatismus zum sicheren Verteilen dieser liefert, müssen diese manuell durch den Administrator auf jeden Client eingestellt werden. Ein häufiger Schlüsselwechsel ist somit aus administrativen Gründen kaum möglich, weshalb ein potentieller Angreifer zeitlich in die Lage versetzt wird, diesen zu entschlüsseln.

Außerdem erschwert diese Art des Schlüsselmanagements die Vergabe von temporären Zugriffsberechtigungen. Möchte ein Unternehmen beispielsweise einen Gast für einen kurzen Zeitraum Zugriff auf das Unternehmens-WLAN gewähren, so muss in jedem Fall der gemeinsame Schlüssel bekannt gegeben werden. Ein Entzug dieser Berechtigung ist jetzt lediglich durch den Austausch aller Schlüssel - sowohl in dem AP als auch in den Clients - möglich.

Abschließend sollen in diesem Zusammenhang mehrere Probleme der Authentifizierung angesprochen werden. Ein einfacher aktiver Angriff, bei dem ein Mitschneiden eines anderen Authentifzierungs-Vorganges vorausgegangen sein muss, kann die Sicherheit der im

WEP-Standard beschriebenen Authentifizierungsmethode Shared-Key ad absurdum führen. Wenn der Angreifer durch Mithören einen vollständigen Challenge-Response-Ablauf eines Clients mit dem AP mitgeschnitten hat, so kann er durch eine XOR-Verknüpfung auf Challenge und Response den Schlüsselstrom errechnen. Mit diesem besitzt der Angreifer die Möglichkeit, zu jeder Challenge eine Response zu erzeugen und sich damit im WLAN zu authentifizieren. Des Weiteren wird im Authentifizierungsprotokoll von WEP nur eine einseitige Authentifizierung vorgenommen, d.h. der Client authentifiziert sich am AP, der Access Point hingegen muss nicht seine Authentizität nachweisen. Somit kann der Client nicht erkennen, ob seine Datenpakete nicht von einem fremden AP, der mit einer stärkeren Sendeleistung den eigentlichen AP überlagert, abgefangen werden.

Zuletzt sei an dieser Stelle noch erwähnt, dass alle Authentifizierungs-Vorgänge nur bezogen auf die Client-Hardware durchgeführt werden. Eine Zugriffskontrolle bzgl. der Benutzer ist in der WEP-Architektur nicht vorgesehen, deshalb können in diesem Sicherheitsmodell auch keine differenzierten Zugriffsrechte vergeben werden.

Neben den an dieser Stelle erwähnten Sicherheitslücken sind in der Literatur viele weitere Schwächen der WEP-Architektur aufgeführt. Nähere Informationen dazu liefern die Literaturquellen [Hag03] sowie [Fre03].

2.5 WPA-Sicherheitsarchitektur

Wie im vorangegangen Kapitel deutlich gemacht wurde, ist die WEP-Sicherheitsarchitektur völlig unzureichend zur Absicherung der WLAN-Kommunikation in einem Unternehmen. Zu dieser Erkenntnis gelangte auch die IEEE und gründete unter der Bezeichnung IEEE 802.11i eine Arbeitsgruppe zur Standardisierung einer höherwertigen Sicherheitsarchitektur. Da jedoch der Standardisierungsprozess der IEEE sehr viel Zeit in Anspruch nimmt, nahm sich auch die WiFi-Allianz dieser Problematik an, um eine Teilmenge der 802.11i Sicherheitsarchitektur sehr kurzfristig unter der Bezeichnung WPA - WiFi Protected Access - als eine vorübergehende Lösung zur Verfügung zu stellen.

Der folgende Abschnitt stellt die WPA-Architektur in Grundzügen vor.

Bei den Bestrebungen der WiFi-Allianz, eine robustere Sicherheitsarchitektur als die bisher im IEEE 802.11-Standard implementierte zu bieten, gab es zwei entscheidende Ziele: zum Ersten musste dieser Pseudostandard sehr schnell verfügbar werden, um kurzfristig eine Alternative zur WEP zu bieten und damit die Verbreitung von WLAN-Geräten weiter zu fördern, zweitens sollte diese Architektur eine Abwärtskompatibilität (in Form von Firmware-Updates) mit den bisher verkauften WEP-fähigen Geräten erlauben und damit Investitionssicherheit gewähren.

Da die Hardwareanforderungen durch die gewünschte Kompatibilität sehr begrenzt waren, konnte kein neuer Verschlüsselungsstandard eingesetzt werden, stattdessen wurde mit einem optimierten Schlüsselmanagement sowie neuen Protokollen und Verfahren die Erreichung der Schutzziele Vertraulichkeit, Integrität sowie Authentizität verbessert.

Schlüsselmanagement Statt einem gemeinsamem Schlüssel zur Verschlüsselung und Authentifizierung kommen bei WPA nur aus einem Master Key abgeleitete kurzlebige Sitzungsschlüssel zum Einsatz. Das Erraten oder Rekonstruieren des Schlüssels wird somit für den Angreifer unsinnig, da während des Angriffes der Schlüssel bereits mehrmals gewechselt worden sein kann.

Trotzdem muss auch bei der Verwendung von WPA ein gemeinsamer Schlüssel auf allen Clients sowie dem AP existieren, der dann die Grundlage für die Erstellung der Sitzungsschlüssel darstellt. Im einfachsten Fall (vor allem in Heimnetzwerken) wird dieser Master Key manuell in allen beteiligten Geräten eingetragen. Diese als WPA-PSK (Pre-Shared-Keys) bezeichnete Variante ermöglicht jedoch auch wieder Wörterbuchattacken - daher ist hier auf eine geeignete Schlüsselwahl zu achten. Als Pre-Shared Keys (PSK) können 8 bis 63-stellige Zeichenketten verwendet werden.

Ein wesentlich robusteres Verfahren zur Übertragung des Master Keys stellt die „Managed Key"-Variante dar. Hierbei erfolgt die Authentifizierung und der Schlüsselaustausch über den Authentifizierungsstandard IEEE 802.1x (siehe Kapitel 2.7).

Verschlüsselung Während im WEP-Standard die Verschlüsselung optional war, ist bei der Verwendung von WPA diese zwingend vorgeschrieben. Hierfür kommt das Temporal Key Integrity Protocol (TKIP) zum Einsatz, welches den WEP-Verschlüsselungsalgorithmus derart kapselt, dass fast alle Schwächen beseitigt werden. Dafür werden temporäre Schlüssel und verschiedene Schlüssel-Mix-Funktionen angewendet. Eine weitere Sicherheitsverbesserung stellt die Vergrößerung des Initialisierungsvektors von 24 auf 48 Bit dar. So kann es nicht mehr vorkommen, dass sich der IV bei starkem WLAN-Verkehr nach 3 bis 4 Stunden wiederholt, was bei einer Stromverschlüsselung wie RC4 fatale Folgen für die Sicherheit hätte.

Integrität Ebenfalls Teil des TKIP sind neue Maßnahmen zur Integritätssicherung. Hierbei wird ein Message Integrity Code (MIC) in Form des sog. Michael-Algorithmus[1] angewendet. Dieser kann jedoch aufgrund der Leistungseinschränkungen aus Kompatibilitätsgründen keine vollständige Sicherheit gewährleisten, weshalb mit WPA ein Konzept der Gegenmaßnahmen eingeführt wurde. So ist bei einem MIC-Fehler davon auszugehen, dass es sich um einen Angriff handelt. Im Standard ist eine MIC-Fehlerrate von unter zwei Fehlern pro Minute festgelegt. Dies impliziert, dass die Station für 60 Sekunden die Annahme aller weiteren Datenpakete verweigert, wenn innerhalb einer Minute zwei MIC-Fehler aufgetreten sind. Des Weiteren wird bei einem solchen Vorfall ein Wechsel des Sitzungsschlüssels empfohlen.

Authentifizierung Bei der Authentifizierung im WPA-Verfahren wird der Standard IEEE 802.1x „Port Based Access Control" angewendet. Dieses Konzept kontrollierter Netzwerkports, das nicht nur auf WLAN Access Points angewendet werden kann, sondern allgemein für den Einsatz in Layer 2 Netzwerk-Komponenten (Ethernet Switch, Bridge usw.) gedacht ist, erlaubt den Zugriff auf kontrollierte Ports nur nach erfolgreicher Authentifizierung. Die Abbildung 2.5 zeigt die prinzipielle Funktionsweise im WLAN-Anwendungsfall.

[1] Vgl. [Hag03]

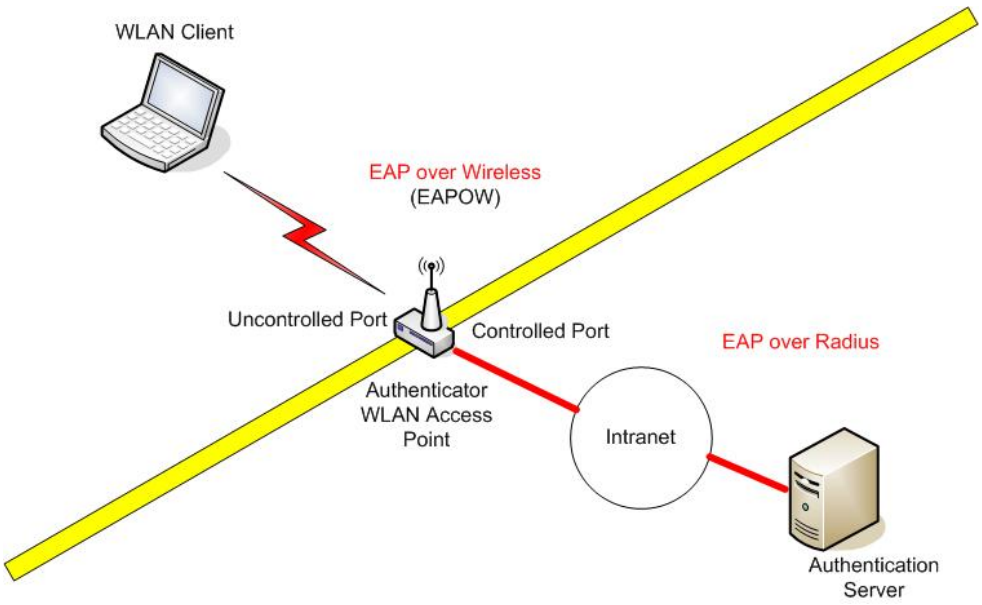

Abbildung 2.5: IEEE 802.1x Port Based Access Control (nach [Ste04])

Der WLAN Client meldet sich über den unkontrollierten Port beim Access Point an, der als Authenticator den Zugang ins Intranet kontrolliert. Zur Verwaltung der Benutzerdaten wird in diesem WPA Enterprise Mode ein zentraler RADIUS-Server (Remote Authentication Dial-In User Service) benötigt. Mit dem Extended Authentication Protocol (EAP), beispielsweise EAP-Transport Layer Security (TLS), wird eine gesicherte Verbindung vom WLAN-Client zum RADIUS-Server aufgebaut. Der Client authentisiert sich auf der Basis eines Client-Zertifikats oder Benutzerpasswortes. Auch der Server muss sich in diesem Ablauf mit Hilfe eines Zertifikates authentisieren - damit können Angriffe durch vorgetäuschte Access Points abgewehrt werden. Bei erfolgreicher Authentisierung erzeugt der RADIUS-Server einen WPA-Schlüssel, der verschlüsselt via EAP-TLS an den Client und an den AP übermittelt wird. Neben EAP-TLS existieren weitere EAP-Verfahren zur gegenseitigen Authentisierung.

In so genannten Small Office / Home Office (SOHO)-Anwendungen, in denen der Betrieb eines speziellen RADIUS-Servers aus Kostengründen nicht möglich ist, kann das bereits angesprochene Verfahren der Pre-Shared-Keys angewendet werden. Dabei erfolgt jedoch der Schlüsselaustausch nicht automatisch, sondern muss manuell durch den Administrator durchgeführt werden.

Eine intensivere Betrachtung der einzelnen Funktionen zur Verschlüsselung und Authentifizierung in der WPA-Sicherheitsarchitektur soll im Rahmen dieser Diplomarbeit nicht durchgeführt werden. Eine ausführliche Darstellung der in „WiFi Protected Access“ genutzten Verfahren befindet sich in [Hag03].

Zusammenfassend lässt sich jedoch feststellen, dass vor allem durch die dynamische Schlüsselverwaltung viele Defizite der WEP-Architektur ausgelöscht werden konnten und damit WPA eine robuste Architektur zur Absicherung von WLAN-Verbindungen ermöglicht. Die einzig bekannten Angriffe nutzen schwache Passwörter - entweder als Pre-Shared-Keys oder

beim Zugriff auf den RADIUS-Server. Diese Sicherheitslücken lassen sich jedoch durch eine geeignete, möglichst zufällige Passwortwahl, schließen.

Da die WPA-Architektur nur eine Teilmenge des im Juni 2004 ratifizierten IEEE 802.11i-Standards ist, soll auch dieser im folgenden Kapitel kurz vorgestellt werden. Vor allem durch die Einbindung des Advanced Encryption Standard (AES) soll er eine noch sicherere Verschlüsselung gewährleisten, allerdings wird dabei die Abwärtskompatibilität mit bisherigen WiFi-Geräten nicht mehr vollständig ermöglicht.

2.6 Der IEEE 802.11i Standard

Um endgültig und langfristig die Sicherheitsproblematik im Wireless LAN zu lösen, wurde von der IEEE eine Task Group mit der Bezeichnung 802.11i gegründet. Dieser lang erwartete Standard ist nun seit Juni 2004 ratifiziert, die ersten kompatiblen Produkte sind seit dem Ende des Jahres 2004 erhältlich. Wie im vorigen Abschnitt erwähnt, hat WPA bereits eine ganze Reihe von Konzepten in 802.11i vorweggenommen - deshalb sollen in diesem Abschnitt nur die Komponenten beschrieben werden, die gegenüber WPA neu sind.

Während die WPA-Sicherheitsarchitektur aus Kompatibilitätsgründen die in ihrer Anwendung unsichere RC4-Verschlüsselung verwendet und damit auch die Nutzung älterer Hardware erlaubt, wird in dem 802.11i-Standard mit CTR/CBC-MAC Protocol (CCMP) ein speziell für WLANs geschaffenes, völlig neues Verfahren eingeführt.

Hierbei wird der AES-Verschlüsselungsstandard mit dem Operationsmodus „Counter Mode (CTR)“ eingesetzt; in der Kombination mit dem Cipher Block Chaining - Message Authentication Code (CBC-MAC), welcher für die Integritätssicherung verantwortlich ist, bildet dieser den CCM - (CTR/CBC MAC)-Mode - und ist damit Grundlage für das im Standard 802.11i spezifizierte Verfahren CCMP.

Die Sicherheit des AES-Algorithmus ist nicht zuletzt wegen seiner Auswahl zum amerikanischen Sicherheitsstandard durch das National Institute of Standards and Technology (NIST) unumstritten, jedoch macht seine Komplexität ein Softwareupdate für ältere WLAN-Chipsätze unmöglich. Daher sieht der Standard 802.11i optional auch die Verwendung von TKIP vor, um eine Kompatibilität mit Altprodukten zu gewährleisten. Bei der Implementation von Geräten nach 802.11i ist jedoch der Einsatz von AES obligatorisch, TKIP hingegen stellt nur eine Option dar und sollte in sicherheitskritischen Anwendungen abgeschaltet werden.

Der Zusatz CCM bezieht sich hierbei auf die Art, wie AES auf WLAN-Pakete angewendet wird. AES ist im Gegensatz zu RC4, welcher eine Stromchiffre realisiert, ein Block-Chiffrier-Algorithmus, d.h. es wird immer gleich ein ganzer Block von Daten (im Falle von AES-CCM 16 Byte) am Stück verschlüsselt. Dabei wird AES sowohl für die Erzeugung des MIC als auch für die Verschlüsselung im „Chained Mode“ angewendet, d.h. für jeden AES-Datenblock werden entweder veränderte Eingangsdaten oder die Ergebnisse des letzten Schrittes verwendet, so dass gleiche Klartextdaten von Schritt zu Schritt einen unterschiedlichen Schlüsseltext ergeben.

Für die Kombination aus Verschlüsselung und Integritätssicherung benötigt AES im Gegensatz zu TKIP nur einen 128 Bit langen Schlüssel, der aufgrund der Symmetrie des Verfahrens in beiden Kommunikationsrichtungen angewendet werden kann. Eine konforme TKIP-Implementierung hingegen verlangt die Verwendung unterschiedlicher Michael-Schlüssel in Sende- und Empfangsrichtung, so dass CCM in seiner Anwendung deutlich unkomplizierter ist als TKIP.

Die Länge des Initialisierungsvektor (IV) beträgt wie bei TKIP 48 Bit, eine IV-Wiederholung ist damit praktisch ausgeschlossen. Zur Vermeidung von Replay-Attacken merkt sich der Empfänger den zuletzt benutzten IV und verwirft Pakete mit einem IV, der gleich oder niedriger als der Vergleichswert ist.

Neben der Einführung des Advanced Encryption Standard zur Sicherung der Vertraulichkeit und der Integrität wurden noch Technologien für einen schnellen Funkzellenwechsel, Pre-Authentification und PMK-Caching (Pairwise Master Key) genannt, in den Standard übernommen, die jedoch an dieser Stelle nicht näher erläutert werden sollen. Die Authentifizierung erfolgt wie in der WPA-Architektur mittels IEEE 802.1x über EAP.

Somit ergibt sich durch die Einführung von 802.11i vor allem eine Verbesserung der Sicherheitsfunktionen im Wireless LAN - ob hierfür die im Interimstandard WPA vorgestellte TKIP-Verschlüsselung nicht ausgereicht hätte, ist aufgrund von fehlenden erfolgreichen Angriffen bisher nicht geklärt. Jedoch schätzen einige US-Behörden diese schwächere Verschlüsselungsvariante als unsicher ein. Erst mit AES wurde somit der endgültige Schritt zu einem aus heutiger Sicht wirklich sicheren Verschlüsselungsverfahren vollzogen, das die bekannten praktischen und theoretischen Sicherheitslücken der Vorgängerversionen schließen kann. Damit werden auch die von vielen US-Behörden zu beachtenden Sicherheitsvorschriften des Federal Information Processing Standards (FIPS) 140-2 eingehalten.

Für die auf IEEE 802.11i basierende Sicherheitsarchitektur werden häufig die Begriffe WPA2 bzw. Robust Secure Network (RSN) synonym verwendet.

2.7 Das Authentifizierungsprotokoll IEEE 802.1x

Sowohl die WPA-Sicherheitsarchitektur als auch deren Nachfolgestandard WPA2, definiert in der Norm IEEE 802.11i, nutzen zur Authentifizierung das Authentisierungsprotokoll IEEE 802.1x, welches wiederum das Extensible Authentication Protocol (EAP) anwendet. Die Funktionsweise und Anwendung dieser beiden Protokolle sollen nun im Detail vorgestellt werden.

2.7.1 Grundlagen von IEEE 802.1x

Der Standard IEEE 802.1x wurde Mitte 2001 verabschiedet, um auf Ebene der Sicherungsschicht im ISO/OSI-Modell eine Zugriffskontrolle ermöglichen zu können. Im speziellen wird damit ein Protokoll in der Familie der IEEE 802-Netze vorgeschlagen, das Netzwerkanschlüsse (Ports) nur berechtigten Benutzern zugänglich machen soll. Hierfür bietet der

Standard die Möglichkeiten der Authentifizierung und Autorisierung. Vor allem im Umfeld der drahtlosen Netze ergibt sich aus der Verwendung von 802.1x ein weiterer Vorteil: der Austausch von Sitzungsschlüsseln zwischen Client und Access Point kann automatisiert erfolgen.

Die Art und Weise der Authentifizierung (Passwort, Challenge-Response, Zertifikat) wird im Standard offen gehalten. Vielmehr wird die Kontrolle der Ports (gesperrt, geöffnet) definiert und ein Paketformat festgelegt, in dem beliebige Authentifizierungsprotokolle gekapselt werden können.

Die folgenden Begriffe werden durch IEEE 802.1x definiert:

Authenticator Eine Einheit an einem Ende eines Punkt-zu-Punkt LAN-Segments, welche die Einheit am anderen Ende der Verbindung authentisiert. Somit stellt der Authenticator den Netzwerkzugang zur Verfügung.

Supplicant Eine Einheit am Ende eines Punkt-zu-Punkt LAN Segments, die bei Zugriffswunsch auf den Netzwerkzugang durch den Authenticator authentifiziert werden muss.

Authentication Server Eine Einheit, die einem Authenticator einen „Authentication Service“ bereitstellt. Anhand der vom Supplicant zur Verfügung gestellten Informationen (Passwort, Zertifikat etc.) bestimmt dieser, ob der Zugriff auf das Netzwerk gewährt werden darf.

Network Access Port Ein Verbindungspunkt zu einem LAN. Dieser kann sowohl physisch (z.B. Netzwerkdose) als auch logisch (z.B. WLAN-Assoziierung) vorhanden sein.

Port Access Entity (PAE) Die einem Port zugeordnete Kontrolleinheit. Die PAE kann die Rolle des Supplicants, des Authenticators oder beide Übernehmen.

2.7.2 Authentifizierung mit IEEE 802.1x

Die PAE des Authenticators ist zweigeteilt: Sie verfügt über einen Controlled Port und einen Uncontrolled Port. Der Controlled Port stellt den allgemeinen Netzwerkzugang zur Verfügung und ist daher vor unberechtigter Benutzung zu sichern. Vor der Authentisierung des Supplicants befindet er sich daher in einem gesperrten Zustand. Der Uncontrolled Port ist ständig geöffnet, erlaubt jedoch nur den Datenverkehr, der zur Authentisierung des Supplicants notwendig ist. Pakete, die nicht der Authentisierung dienen, werden verworfen.

Der Port des Authenticators unterscheidet zwischen zwei Zuständen: Entweder ist der Supplicant autorisiert, dann ist der Port geöffnet, oder der Supplicant ist nicht autorisiert, dann ist der Port geschlossen. Zunächst ist der Supplicant nicht autorisiert und der Controlled Port ist gesperrt, so dass kein Datenaustausch mit dem Netzwerk stattfinden kann. Nach erfolgreicher Authentisierung wird der Controlled Port geöffnet und der Client hat Zugriff auf das Netzwerk.

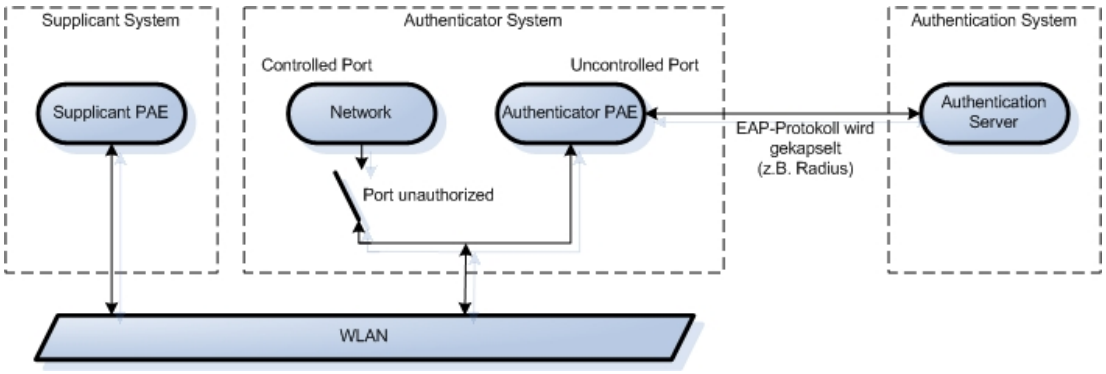

Abbildung 2.6: Supplicant, Authenticator und Authentication Server im Protokoll IEEE 802.1x (nach [Ive03])

Ob ein Supplicant autorisiert ist einen Netzwerkzugang zu erhalten, entscheidet der Authentication Server. Der Authenticator befindet sich zwischen Supplicant und Authentication Server und vermittelt zwischen ihnen, wie in Abbildung 2.6 gezeigt.

Möchte sich ein Supplicant zur Erlangung des Netzwerkzugangs authentifizieren, schickt er ein Authentifizierungs-Request an den Authenticator. Dieser verfügt über eine Netzwerkverbindung zum Authentication Server, worüber er die Authentifizierungsanfrage weiterleitet. Der Authentication Server - in den meisten Fällen ein Dienst, der das Remote Authentication Dial-In User Service (RADIUS)-Protokoll implementiert - prüft nun die Autorisierung des Supplicants und teilt das Ergebnis dem Authenticator mit. Bei einer positiven Autorisierung öffnet dieser den Controlled Port und erlaubt somit dem Supplicant den Netzwerkzugang.

Die eigentliche Logik hinter der Authentizierung muss nicht im Authenticator implementiert sein. Gemäß dem Standard nimmt der Authenticator nur eine Vermittlerrolle zwischen Supplicant und Authentication Server ein. Die Kommunikation zwischen Authenticator und Authentication Server findet auf einem Protokoll höherer Ebene statt (zumeist RADIUS über TCP/IP), während Supplicant und Authenticator mit „EAP over LAN“ kommunizieren. Der Authenticator verarbeitet dabei nur bedingt den Inhalt der Pakete, die Authentication Server und Supplicant austauschen. Nach erfolgreichem Abarbeiten des Authentifizierungsprotokolls schickt der Authentication Server ein EAP-Success-Paket an den Authenticator, der dann den Controlled Port für den Supplicant öffnet und diesem das EAP-Success-Paket übermittelt.

Nachdem die Authentifizierung erfolgt ist, nutzt der Supplicant das Netz mit den gewohnten Netzwerkprotokollen. Bis auf die eigentliche Authentifizierung erzeugt 802.1X keinen Paket-Overhead.

2.8 Das Extensible Authentication Protokoll (EAP)

Innerhalb des Authentifizierungs-Vorganges nimmt EAP eine bedeutende Stellung ein, da es über die Art und Weise der Authentifizierung entscheidet. Daher soll an dieser Stelle das Protokoll sowie die darauf bestehenden Verfahren näher erläutert werden.

Hintergrund der Standardisierung von EAP durch die Internet Engineering Task Force (IETF) ist die gestiegene Anzahl von Authentisierungsverfahren für PPP, das Point-to-Point-Protokoll. Das Ziel des Standards war die Festlegung einer Rahmenstruktur, in welches jedes beliebige Authentisierungsverfahren gekapselt werden kann, um somit dem PPP-Protokoll einen flexiblen Authenfizierungsmechanismus bereitstellen zu können.

2.8.1 Aufbau eines EAP-Paketes

Ein EAP-Paket besteht aus vier Feldern:

Bytes	1	1	2	Variabel
	Code	Identifier	Length	Data

Tabelle 2.1: Aufbau EAP-Paket

Code Das Code-Feld mit der Größe eines Bytes identifiziert den Typ des EAP-Pakets und gibt somit an, wie das Datenfeld zu interpretieren ist.

Identifier Das Identifier-Feld enthält eine Zahl vom Typ „unsigned int“, um zusammengehörende Request und Responses zu identifizieren. Bei einer Wiederholung der Übertragung bleibt der Identifier gleich, ansonsten wird jeweils ein neuer Identifier verwendet.

Length Das zwei Bytes große Length-Feld enthält die Gesamtlänge des EAP-Paketes.

Data Das Datenfeld, welches von variabler Länge ist, enthält die für die Authentifizierung notwendigen Nutzdaten. Die Interpretation ist dabei abhängig vom Wert des Code-Feldes.

Es existieren vier verschiedene Arten von EAP-Paketen: Request, Response, Success und Failure.

Zu Beginn der Authentifizierung sendet der Supplicant die Request-Daten an das System, um Zugang zu erhalten. Dabei wird das Datenfeld zweigeteilt, um den Typ des Requests und die dazu gehörigen Daten zu übermitteln. Auch die EAP-Response nutzt dieses Paket-Format, um Antworten vom Authenticator an den Supplicant zu verschicken.

Die Tabelle 2.2 zeigt die bisher spezifizierten Typen von EAP-Anfragen und -Antworten.

Der Request for Comments (RFC) 2284 sieht vor, dass jede Implementierung von EAP die Typen 1-4 unterstützt, wobei lediglich Typ 4 ein Authentisierungs-Protokoll ist. Typ 1 ist reserviert, um die Identität des Gegenübers zu erfragen. Anfragen des Typs 2 (Notification) enthalten einen String, der den User über den Ablauf seines Passwortes o.ä.

1	Identify
2	Notification
3	NAK (new authentication method)
4	MD-5 Challenge
5	OTP (One-time Password)
6	Generic Token Card
13	TLS (Transport Layer Security)
17	EAP Cisco Wireless (LEAP)
21	EAP-Tunneled TLS (TTLS)
25	Protected EAP (PEAP)

Tabelle 2.2: EAP-Codes

informiert. Typ 3 (NAK) kommt nur in Responses vor und wird geschickt, wenn der Authentifizierende ein Request nicht beantworten kann, weil etwa das Authentifizierungsprotokoll nicht implementiert ist, oder weil er es nicht einsetzen will. EAP-NAKs enthalten in der 'Type-Data' einen 8-bit Wert, der ein Protokoll zur Authentifizierung vorschlägt.

2.8.2 Beispiel zum EAP-Austausch

1. Der Authenticator sendet Request/Identity Paket, um den Benutzer zu identifizieren.
2. Der Supplicant antwortet mit seiner ID.
3. Daraufhin sendet der Authenticator eine Anfrage, das MD-5 verwendet werden soll.
4. Das Benutzer-System ist auf EAP-PEAP konfiguriert und antwortet daher mit einem NAK mit Vorschlag auf EAP-PEAP.
5. Der Authenticator fragt also nach einer EAP-PEAP Authentifizierung.
6. Der Benutzer antwortet mit einem falschen PEAP Passwort.
7. Da der Authenticator das Passwort als falsch erkennt, er aber mehrfache Authentifizierung implementiert hat, fragt er erneut nach.
8. Dieses Mal antwortet der Benutzer mit einem gültigen Passwort.
9. Die Antwort ist korrekt, worauf der Authenticator eine Success-Nachricht sendet.

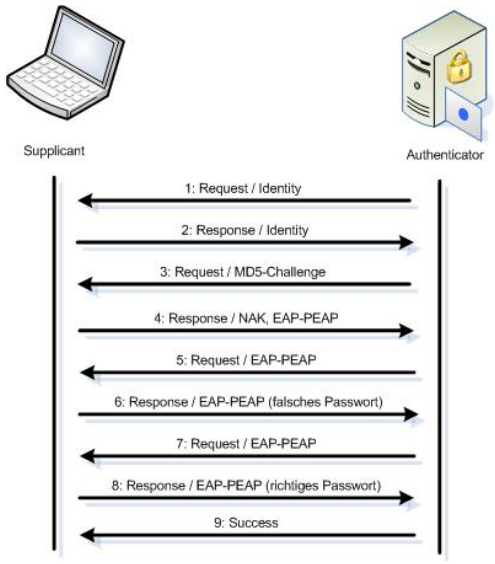

Abbildung 2.7: Beispiel EAP-Austausch

2.8.3 EAP over LAN (EAPOL)

Da während der Authentifizierung zwischen Supplicant und Authenticator auf kein verhältnismäßig komfortables Protokoll wie das Internet Protocol (IP) zurückgegriffen wer-

den kann, müssen EAP-Pakete direkt in den Nutzdaten von Ethernet- oder Wireless-LAN-Paketen übermittelt werden. Das dafür verwendete Protokoll wird als EAP over LAN (EAPOL), im WLAN-Umfeld auch häufig EAP over Wireless (EAPOW) bezeichnet. Neben der Übermittlung von EAP-Paketen bietet dieses Protokoll die notwendigen Erweiterungen zum Austausch dynamischer Schlüsselinformationen.

Eine Übersicht über die verschiedenen EAPOL-Pakete befindet sich in der Tabelle 2.3.

Nummer	**Name**
0	EAP-Paket
1	EAPOL-Start
2	EAPOL-Logoff
3	EAPOL-Key
4	EAPOL-Encapsulated-ASF-Alert

Tabelle 2.3: EAPOL-Typen

EAPOL-Start und EAP-Logoff-Pakete werden vom Supplicant an den Authenticator versandt, um den Beginn einer Authentifizierung zu erbitten bzw. um nach einer Nutzung eines Controlled Ports das Ende der Benutzung bekannt zu geben.

Eine große Bedeutung für die Anwendung von EAPOL im WLAN-Umfeld besitzt das Paketformat EAPOL-Key. Über dieses können Schlüsselinformationen zwischen Supplicant und Authenticator ausgetauscht werden, aus welchen dann dynamische Sitzungsschlüssel für die WLAN-Übertragung generiert werden.

Das EAPOL-Paket vom Typ 4 dient der Übertragung von Alarmen und Statusinformationen an einen unautorisierten Port.

Die bisher beschriebenen Protokoll EAP und EAPOL stellen jedoch nur ein Framework für die Übertragung von Authentifizierungsinformationen dar. Für die eigentliche Authentifizierung sind gesonderte Verfahren notwendig, welche in den folgenden Abschnitten beschrieben werden. Aus der großen Anzahl von EAP-Verfahren sollen jedoch nur solche diskutiert werden, die Schlüsseldaten generieren und somit für die Verwendung in drahtlosen Netzwerken genutzt werden können.

2.8.4 EAP-TLS

Das EAP-TLS Verfahren, veröffentlicht im RFC 2716, wurde entwickelt, um Transport Layer Security (TLS) unter EAP zur Verfügung zu stellen.

Die Authentifizierung stützt sich dabei auf X.509-Zertifikate[2], welche sowohl für den Client (Supplicant) als auch für den Server (Authentication Server) vorhanden sein müssen. Der Server übermittelt dabei sein Zertifikat an den Client, welcher es auf Vertrauenswürdigkeit überprüft. Außerdem authentifiziert sich der Client in umkehrter Richtung durch

[2] X.509 ist ein ITU-T-Standard für eine Public Key-Infrastruktur und ist derzeit der wichtigste Standard für digitale Zertifikate.

Vorlage seines Zertifikates beim Server. Nach erfolgreicher gegenseitiger Authentifizierung und Aushandelung eines Sitzungsschlüssels sowie eines Verschlüsselungsverfahrens ist die EAP-TLS Authentifizierung beendet. Der dabei ausgehandelte Sitzungsschlüssel ist sicher, da TLS garantiert, dass während der Authentifizierung keine Pakete unentdeckt abgeändert, unberechtigt erzeugt bzw. mitgelesen werden können.

Der Vorteil dieses Verfahrens liegt in der gegenseitigen Authentifizierung und dem sicheren Schlüsselaustausch zwischen beiden Endpunkten.

Jedoch fordert EAP-TLS die Existenz einer Public-Key-Infrastruktur und damit die sichere Erstellung und Verteilung von Zertifikaten sowohl auf Server-, als auch auf Client-Seite. Außerdem lässt sich mit EAP-TLS eher eine Maschinenauthentifizierung als eine Benutzerauthentifizierung durchführen, da aufgrund der Speicherung des Clientzertifikates auf einem PC jeder Zugriffsberechtigte das Zertifikat nutzen, aber auch kopieren kann. Somit kann nicht sichergestellt werden, dass der Benutzer des Zertifikates identisch ist mit dem Besitzer des Clientzertifikates. Mehr Sicherheit kann hier durch die Abfrage von Benutzernamen und Passwörtern erreicht werden, die in den folgenden EAP-Methoden implementiert ist.

2.8.5 EAP-TTLS

Eine Erweiterung des EAP-TLS-Verfahrens wird mit dem Protokoll EAP-Tunneled Transport Layer Security (TTLS) eingeführt. Zu Beginn der Authentifizierung erfolgt dabei basierend auf dem Server-Zertifikat der Aufbau eines sicheren TLS-Tunnels zwischen WLAN-Client und Authentisierungsserver. In diesem sicheren Tunnel kann sich nun der Benutzer mittels einer Kombination von Benutzername und Passwort beim Authentisierungsserver identifizieren. Die umständliche Notwendigkeit eines Benutzerzertifikates entfällt hierbei - es wird wie bei SSL/TLS Authentisierungen im Internet üblich nur ein serverseitiges Zertifikat benötigt. Das EAP-TTLS-Verfahren übernimmt somit die Vorteile von EAP-TLS und beseitigt die aufwendige Verwaltung und Verteilung von Benutzerzertifikaten.

Innerhalb des TLS-Tunnels kann nun wieder zwischen einer Vielzahl von Protokollen zur Übertragung des Benutzernamens und des Passworts gewählt werden. Entweder verwendet man innerhalb des Tunnels wiederum ein EAP-Protokoll, oder man nutzt hierbei ältere Authentifizierungen wie das Password Authentication Protocol (PAP), Challenge Handshake Authentication Protocol (CHAP) oder die Microsoft-spezifische Variante MS-CHAPv2, die an dieser Stelle nicht näher erläutert werden sollen.

2.8.6 PEAP

Protected EAP (PEAP) basiert ebenfalls auf TLS und benutzt daher einen sicheren Tunnel zur Übertragung von Benutzername und Passwort. PEAP ist zwar in Funktionalität und Sicherheit mit EAP-TTLS vergleichbar, aber nicht so flexibel. Die Unterstützung für die Authentifizierung im Tunnel ist außerdem nicht so weitreichend wie bei EAP-TTLS. Kommerzielle Implementierungen von PEAP, die seit Anfang 2003 auf dem Markt sind,

zeigten deutliche Interoperabilitätsprobleme. Dennoch wird für dieses Protokoll aufgrund der Unterstützung durch Microsoft und Cisco eine große Akzeptanz erwartet. Das Protokoll eignet sich daher perfekt für eine sichere Authentifizierung gegenüber Windows-Domänen und Verzeichnisdiensten.

2.8.7 Zusammenfassung und Vergleich der EAP-Methoden

Die folgende Tabelle stellt die verschiedenen, im WLAN-Umfeld sinnvollen EAP-Methoden gegenüber:

Verfahren	**Authentifizierung**		**Standard**
	Von	Über	
EAP-TLS	Client / Server	Zertifikate	RFC 2716
EAP-TTLS	Client Server	Passwort o.ä. Zertifikat	draft-ietf-pppext-eap-ttls
PEAP	Client Server	MS-CHAP-Passwort o.ä. Zertifikat	draft-josefsson-pppext-eap-tls-eap

Tabelle 2.4: EAP-Methoden im Vergleich

Allen genannten Methoden gemeinsam ist die Möglichkeit zur dynamischen Schlüsselgenerierung sowohl für WEP, WPA als auch WPA2 / 802.11i. Während EAP-TLS über ein RFC bereits hinreichend spezifiziert ist, liegen EAP-TTLS und PEAP im Augenblick nur als Draft vor und können daher noch verändert werden.

Somit ist die Auswahl einer geeigneten EAP-Methode jeweils abhängig von der vorhandenen Infrastruktur, also dem eventuellen Vorhandensein einer Public-Key-Infrastruktur (PKI), und der Frage der Verfügbarkeit von Implementierungen sowohl auf Client- als auch auf Server-Seite.

2.9 Sicherung von WLANs durch IPSec

Neben den bisher beschriebenen Sicherheitsarchitekturen, die alle direkt für Wireless LAN entwickelt wurden, existiert mit der Norm Internet Protocol Security (IPSec) eine weitere Möglichkeit zur Absicherung von WLAN-Verbindungen. IPSec ermöglicht, bei IP-Datagrammen die Authentizität des Absenders, die Vertraulichkeit und die Integrität der Daten zu wahren. Weiterhin existiert in IPSec auch ein Schutz vor Replay-Attacken, also dem wiederholten Senden alter Datagramme. Diese Aufgaben werden durch eine Erweiterung der Datagramme um optionale Header erfüllt.

Bis zum Aufkommen der WLAN-Technologie wurde der Standard IPSec bisher nur zum Aufbau sogenannter Virtueller Privater Netzwerke VPN genutzt. Dabei können entfernte private Netzwerke über eine öffentliche Infrastruktur, die als Transitnetz verwendet wird, zusammengeschlossen werden. Die zwischen den privaten Netzen verschickten Datenpakete werden im Tunnel-Modus über die unsichere Verbindung geschickt, weshalb zur Sicherung

der Vertraulichkeit, Integrität und Authentizität die Anwendung von kryptographischen Verfahren notwendig ist.

Aufgrund der fehlenden Verfügbarkeit robuster Sicherheitsarchitekturen in Wireless LANs wird das IPSec-Verfahren auch in WLAN-Umgebungen, vor allem im Unternehmensumfeld häufig eingesetzt. Dabei können die in IEEE 802.11 mitgelieferten Sicherheitsfunktionen (in Altgeräten zumeist nur WEP) deaktiviert werden; alle Maßnahmen durch Absicherung der Kommunikation werden dann softwaremäßig auf Clients und Access Point durch IPSec realisiert.

Da IPSec in der Vermittlungsschicht (Schicht 3) des ISO/OSI Schichtenmodells implementiert wird, können alle Anwendungen, die mit TCP/IP arbeiten, trotz IPSec weiter verwendet werden. Für sie und ihre Benutzer ändert sich nichts, da sie in höheren Schichten agieren. Daher ist IPSec für die Anwendungen und ihre Benutzer völlig transparent. Aufgrund der Standardisierung von IPSec durch die IETF sowie der Veröffentlichung der dazugehörigen Protokolle in einer Reihe von RFC's existiert neben einigen herstellerspezifischen Lösungen auch eine genügende Anzahl von Open Source Produkten.

In den nun folgenden Kapiteln sollen die unterschiedlichen Modi von IPSec sowie die dabei eingesetzten Protokolle erläutert werden.

2.9.1 Transport Mode und Tunnel Mode

Für den Betrieb von IPSec stehen zwei Modi zur Auswahl: Transport Mode und Tunnel Mode.

Im Transport Mode wird in das IP-Datagramm zwischen den IP-Header und den TCP- oder UDP-Header ein IPSec-Header eingesetzt. Da der IP-Header unverändert bleibt, eignet sich dieser Modus nur für Ende-zu-Ende-Verbindungen. Die folgende Abbildung zeigt ein solches erweiteres Datagramm.

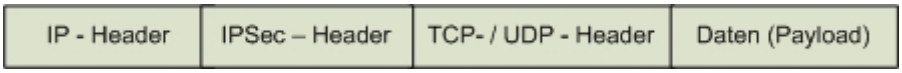

Abbildung 2.8: IP-Datagramm im Transport Mode

Der Tunnelmodul bietet mehr Flexibilität, denn hierbei wird dem gesamten IP-Datagramm ein IPSec-Header und ein neuer IP-Header vorangesetzt. Abbildung 2.9 illustriert das gesamte IP-Datagramm im Tunnel Mode.

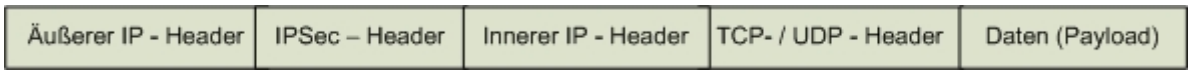

Abbildung 2.9: IP-Datagramm in Tunnel Mode

Damit ist es möglich, eine Ende-zu-Gateway- oder eine Gateway-zu-Gateway-Verbindung aufzubauen. Der äußere IP-Header enthält die IP-Adressen der beiden Gateways und

adressiert damit die kryptographischen Endpunkte. Der innere IP-Header enthält die IP-Adressen des tatsächlichen Ziels und der tatsächlichen Quelle.

Für die Sicherung der Daten existieren in IPSec zwei Protokolle: Authentication Header (AH) und Encapsulation Security Payload (ESP). Beide können sowohl im Transport Mode als auch im Tunnel Mode genutzt werden.

2.9.2 Authentication Header

Durch das Protokoll AH können in IPSec die Sicherheitsziele Authentizität, Integrität und Schutz gegen Replay-Angriffe erreicht werden. Eine Verschlüsselung der Daten findet dabei nicht statt.

Das Authentication Header Protokoll erzeugt für jedes Datenpaket einen zusätzlichen Header. Dieser Header enthält die notwendigen Informationen, um eine Authentifizierung durchzuführen. Die Forderungen Authentizität und Integrität werden mittels eines Hashwertes überprüft, der durch einen Hashalgorithmus erzeugt worden ist. Es stehen zwei Hashalgorithmen zur Auswahl:

- HMAC-MD5, erzeugt einen Hashwert mit 128 Bit Länge
- HMAC-SHA, erzeugt einen Hashwert mit 160 Bit Länge

Der Replay-Angriff wird durch die Angabe der Folgenummer verhindert, denn damit kann der Empfänger erkennen, ob ein Datenpaket wiederholt gesendet wurde.

Der Authentifikationsbereich, also der Bereich, der durch die Anwendung des Hashwertes gesichert wird, erstreckt sich, wie in Abbildung 2.10 gezeigt, über das gesamte Datenpaket.

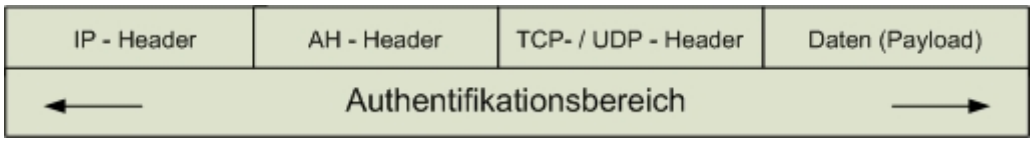

Abbildung 2.10: IP-Datagramm mit AH im Transport Mode

Obwohl der AH-Header mittels des Hashwertes das ganze Datenpaket abdeckt, gibt es einige variable Felder (Bsp. Time to Live, Header Checksum, usw.), die im Hashwert nicht berücksichtigt werden, da sie beim Transport vom Ursprungsort zum Zielort verändert werden und somit den Hashwert ungültig machen würden.

2.9.3 Encapsulation Security Payload

Wenn bei der Datenübertragung mit IPSec das Schutzziel Vertraulichkeit gefordert ist, kann das AH-Protokoll nicht angewendet werden. Für IPSec-basierte Übertragung auf unsicheren Kanälen wird daher das ESP-Protokoll verwendet, welches zusätzlich zu den in AH realisierten Sicherheitsfunktionen eine Verschlüsselung durchführt. Durch das zusätzliche Verschlüsseln werden die Informationen vor unberechtigten Lesern geschützt. Folgende Verschlüsselungsalgorithmen werden dabei zumeist angewendet:

DES	Data Encryption Standard
3DES	Triple Data Encryption Standard
IDEA	International Data Encryption Standard
AES	Advanced Encryption Standard

Im ESP-Modus wird im Gegensatz zum AH-Modus die IP-Adresse nicht in die Authentifizierung einbezogen. Das im Transport Mode mit ESP generierte Datenpaket ist folgendermaßen aufgebaut:

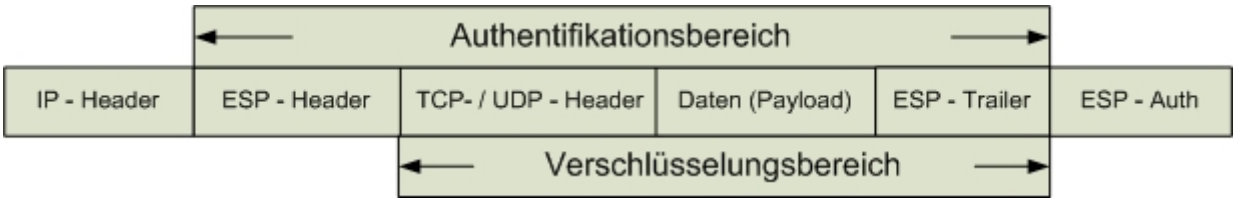

Abbildung 2.11: Datagramm mit ESP im Transport Mode

Die Vertraulichkeit wird mit symmetrischen Verschlüsselungsalgorithmen im CBC-Modus realisiert. Dabei wird alles zwischen ESP-Header und den Authentifikationsdaten verschlüsselt. Im Standard ist die Implementation von DES obligatorisch, weitere Algorithmen können optional integriert werden.

Mit Hilfe von HMACs wird die Integrität und Authentifikation des Absenders realisiert. Als HMAC wird eine kryptographisch sicherere Variante der Message Authentication Codes (MAC) bezeichnet, welche in einem zweistufigen iterativen Verfahren mit Hilfe von Hashfunktionen eine Prüfsumme erstellt. HMACs, definiert im RFC 2104, stellen weitgehend sicher, dass eine Prüfsumme genau auf einen Text abgebildet wird. Es ist mit der heutigen Technik unwahrscheinlich, zwei unterschiedliche Texte zu finden, die dieselbe Prüfsumme haben. Im ESP-Protokoll wird die Prüfsumme über den Bereich vom ESP-Header bis zum ESP-Trailer erstellt und am Ende des Datenpakets im Bereich Authentifikationsdaten abgelegt. Standardgemäß müssen mindestens HMAC-MD5 und HMAC-SHA implementiert werden, weitere HMACs sind jedoch ebenfalls möglich.

Das ESP-Verfahren lässt sich flexibel einsetzen. Bei hohen Sicherheitsanforderungen können Verfahren für Verschlüsselung und Authentifikation festgelegt werden, wenn eine der beiden Sicherheitsfunktionen nicht benötigt wird (z.B. bei Kommunikation über einen sicheren Kanal) lässt sich jeweils die Verschlüsselung oder die Authentifikation deaktivieren.

Jedoch ist es nicht möglich, auf beide Verfahren zu verzichten, da dann trotz der Verwendung von IPSec keinerlei Schutzmechanismen angewendet werden und das Datenpaket vom Sicherheitsniveau dem eines ungeschützten Datagramms entsprechen würde.

Die Sicherheit des vorgestellten ESP-Protokolls ist bei der Wahl geeigneter Algorithmen unumstritten, jedoch ist auch hier ein Schlüsselmanagement notwendig, um Kommunikationsschlüssel sicher erzeugen und verteilen zu können. Dieses Schlüsselmanagement wird in IPSec durch das Internet Key Exchange (IKE) - Protokoll sicher durchgeführt.

2.9.4 Internet Key Exchange

Der Internet Key Exchange ist ein von IPSec unabhängiger Standard, definiert in den RFCs 2407 bis 2409. IKE basiert auf den Protokollen Internet Security Association and Key Management Protocol (ISAKMP) und Oakley[3].

Die Hauptaufgabe von IKE besteht in der automatischen Realisierung der folgenden drei Komponenten für ein sicheres Schlüsselmanagement:

- Authentifikation des Kommunikationspartners
  IKE bietet vier verschiedene Authenitfikatonsmethoden an, um sicherzustellen, dass man auch wirklich mit der richtigen Person kommuniziert
  - Authentifikation mit Pre-Shared-Key
  - Authentifikation mit Digitaler Signatur
  - Authentifikation mit Public-Key-Verschlüsselung
  - Verbesserte Methode für Authentifikation mit Public-Key-Verschlüsselung

  Im Rahmen dieser Arbeit sollen die Methoden mit Pre-Shared-Key (PSK) und Digitaler Signatur näher erläutert werden.
- Erzeugen der Security Association
- Schlüsselerzeugung und Regenerierung. Während des IKE Vorganges werden mehrere Schlüssel erzeugt.

Die Verschlüsselungsschlüssel und Authentifizierungsschlüssel für die eigentliche Nutzdatenübertragung werden dann aus diesen Schlüsseln abgeleitet. Das Regenerieren der Schlüssel nach einer bestimmten Zeit erhöht zusätzlich den Sicherheitsgrad.

Zur Herstellung einer sicheren Verbindung ist der Aufbau einer so genannten Security Association (SA) notwendig. Diese enthält Informationen über die Kommunikation mit dem anderen Rechner, beispielsweise die verwendeten gemeinsamen Geheimnisse (Schlüssel) sowie die Lebensdauer der SA. Diese SAs bestehen nur während der Kommunikation. Wird erstmals ein IP-Datagramm zu einem Rechner geschickt, für den die IPSec Policy die Absicherung mit IPSec erfordert, wird ein Schlüsselaustauschprotokoll - üblicherweise

[3] Vgl. Requests for Comments (RFC) 2412 [Orm98]

IKE - aufgerufen, um die erforderlichen SAs zu erstellen. Die dabei vereinbarte Lebensdauer der SA gibt an, nach welcher Zeiteinheit oder welcher übertragenen Datenmenge der Rekeying-Vorgang gestartet wird, d.h. eine neue SA erstellt werden muss und damit ein Wechsel des Kommunikationsschlüssels stattfindet.

Der periodische Wechsel des Schlüssels verhindert Angriffe auf die Sicherheit des zugrunde liegenden kryptographischen Verfahrens.

Zum Aufbau der IPSec - Security Association sind zwei Phasen erforderlich:

In der ersten Phase baut IKE die so genannte ISAKMP-SA auf, also diejenige Security Associtaton, die zuständig ist für die Verschlüsselung des ISAKMP-Datenpakets selbst. Sie beschreibt, wie die beiden Kommunikationspartner einander authentifizieren und welches Verschlüsselungsverfahren für IKE angewendet wird. Zum Austausch eines gemeinsamen Geheimnisses für die notwenige Verschlüsselung wird der Diffie-Hellmann-Schlüsselaustausch verwendet, welcher an dieser Stelle nicht näher beschrieben werden soll. Danach läuft bereits jede weitere Kommunikation verschlüsselt ab. Es folgt nun die gegenseitige Authentifikation der Kommunikationspartner. Dafür ist die Verwendung gemeinsamer Geheimnisse (PSK), asymmetrischer Verschlüsselungsverfahren oder digitaler Signaturen vorgesehen. Die Nutzung von Pre-Shared-Keys ist aufgrund des schwierigen Schlüsselaustausches nur in Testumgebungen ratsam. Häufig werden stattdessen digitale Signaturen in Form von X.509-Zertifikaten angewendet, die auch in größeren Netzwerken einfach zu verwalten und zu verteilen sind.

Die zweite Phase erzeugt die IPSec-SA, die auf die Nutzdaten angewendet wird. Hierbei wird unter der Nutzung der Schlüssel der ISAKMP-SA sowie einer Zufallszahl ein Hashwert erzeugt, welcher nun als Schlüssel für die IPSec-SA verwendet werden kann. Somit haben sich die beiden Kommunikationspartner auf einen gemeinsamen Schlüssel und auf ein gemeinsames Vorgehen bei der Sicherung der Datenpakete mit IPSec geeinigt. Es ist sichergestellt, dass der eine Kommunikationspartner die geschützten Datagramme des anderen entschlüsseln, sowie auf Authentizität und Integrität prüfen kann.

2.9.5 Fazit

Zusammenfassend lässt sich feststellen, dass IPSec ähnlich dem IEEE 802.11i-Standard sehr robuste Verfahren zur Verschlüsselung, Authentifizierung und Integritätssicherung anbietet. Zudem findet sowohl bei IPSec als auch bei 802.11i ein periodischer automatischer Schlüsselaustausch statt, der die Sicherheit des Gesamtsystems erheblich verbessert. Somit können beide Sicherheitsarchitekturen vom Sicherheitsniveau als gleichwertig angesehen werden, jedoch bietet IPSec aufgrund seiner Unabhängigkeit von der Netzwerkarchitektur entscheidende Vorteile. Mit Hilfe von IPSec Sicherheitsregeln ist genau definierbar, welche Kommunikationspartner in welcher Art und Weise sicher kommunizieren sollen; außerdem lassen sich hierbei unterschiedliche Sicherheitsregeln für verschiedene Benutzergruppen und Netzwerke definieren. Die Hardwareunabhängigkeit von IPSec, welches im Betriebssystem bzw. in Zusatzsoftware implementiert wird, ermöglicht daher auch die Verwendung von älteren WLAN-Geräten, die keine Unterstützung von WPA bzw. IEEE 802.11i bieten. Somit kann vor allem bei Vorhandensein einer bestehenden WLAN-Infrastruktur die Einführung einer robusten Sicherheitsarchitektur mit IPSec wesentlich günstiger realisiert werden. In der praktischen Realisierung dieser Arbeit soll daher das IPSec-Protokoll als Alternative zu den Sicherheitsarchitekturen nach IEEE 802.11i angeboten werden.

Kapitel 3

Die Intel-Hardwareplattform

Die Embedded Linux Distribution für den in dieser Diplomarbeit zu realisierenden Enterprise WLAN Access Point soll gemäß Aufgabenstellung auf die Verwendung des Netzwerkprozessors Intel IXP425 angepasst werden. Zur Entwicklung und zum Test kommt dabei eine Referenzplattform mit der Bezeichnung IXDPG425 zum Einsatz. Im Folgenden sollen sowohl der Prozessor als auch das Referenzgerät vorgestellt werden.

3.1 Der Intel IXP425 Netzwerkprozessor

Zu Beginn des Jahres 2002 wurde von Intel der auf der XScale-Architektur basierende Netzwerkprozessor IXP425 vorgestellt. Der seit Ende 2002 verfügbare, zu einem Preis von 25 bis 45 US-Dollar[1] erhältliche Prozessor wurde vor allem zum Datentransfer über DSL, Kabel- oder drahtlose Verbindungen entworfen.

Der IXP425 verfügt über einen mit 533 MHz operierenden XScale-Kern sowie drei Network Processor Engine (NPE), die parallel zum Kern unterschiedliche netzwerkspezifische Aufgaben bewältigen können und damit den Kern entlasten.

Zwei dieser NPE (NPE A und B) stellen jeweils eine 10/100-Ethernet-Schnittstelle zur Verfügung, dabei kann NPE B zusätzlich auch für die Beschleunigung kryptographischer Algorithmen (SHA-1, MD5, DES, 3DES, AES) verwendet werden. Die dritte NPE (NPE WAN/Voice) beschleunigt die Kommunikation für Digital Subscriber Line (DSL)- und Voice over IP (VoIP)-Anwendungen.

Zudem besitzt der IXP425 die folgenden, in [Int04a] beschriebenen Schnittstellen:

- eine 33/66 MHz PCI-Schnittstelle
- eine USB 1.1 Geräteschnittstelle
- serielle Ports für Voice-over-IP-Dienste
- serielle Schnittstellen (UART) zur Anbindung externer Geräte
- ein UTOPIA 2-Interface für xDSL-Anwendungen

[1] Der Preis ist abhängig von der Abnahmemenge.

- einen SDRAM-Controller für einen Speicherausbau von bis zu 256 MByte

Dieser Prozessor kann in Temperaturbereichen von -40°C bis +85°C eingesetzt werden und hat dabei nur einen Strombedarf zwischen 1 und 1,5 Watt.

Die folgende Grafik zeigt den Aufbau des Intel IXP425-Netzwerkprozessors:

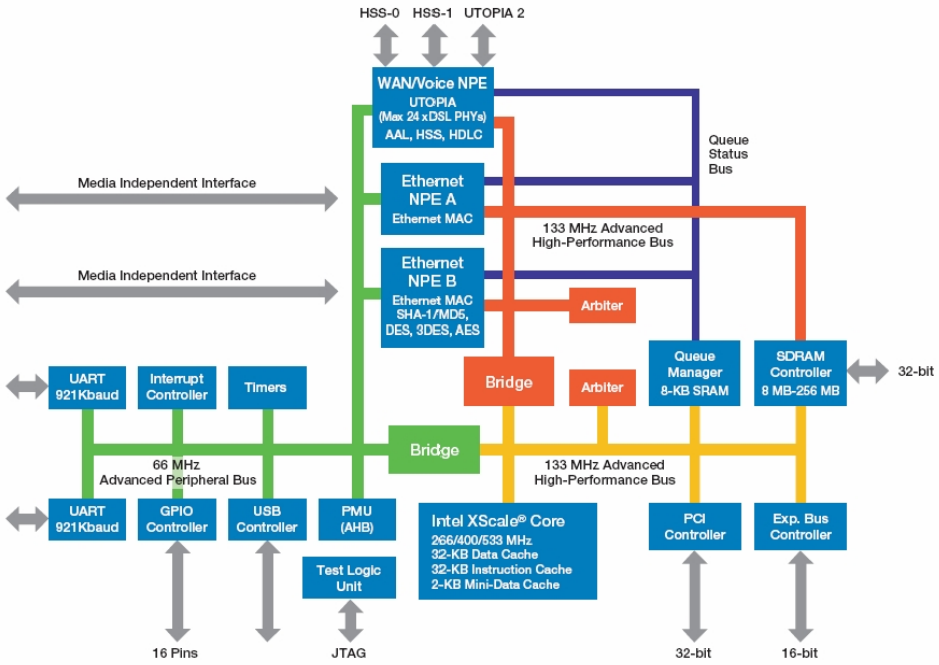

Abbildung 3.1: Aufbau des IXP425 (aus [Int04a])

Zur Entwicklung kundenspezifischer Anwendungen stellt Intel eine auf dem IXP425 basierende Referenzplattform mit der Bezeichnung IXDPG425 zur Verfügung.

3.2 Die Intel Netzwerk-Gateway-Referenzplattform IXDPG425

Für die prototypische Umsetzung des WLAN-Access Points wurde die Intel Referenzplattform IXDPG425 eingesetzt. Dieses Gerät enthält sämtliche Schnittstellen, die zum Aufbau eines Wireless LAN Access Points benötigt werden.

Auf dem Board befinden sich neben dem IXP425-Prozessor ein SDRAM-Speicher von 32 MByte, ein Flash-Speicher von 16 MByte, vier Telefon-Schnittstellen, ein 5-Port Ethernet-Switch von Realtek, eine miniPCI-Schnittstelle für den Anschluss eines WLAN-Adapters, eine serielle Schnittstelle sowie zwei USB 2.0 Host-Schnittstellen.

Den Aufbau des IXDPG425 Evaluation Board zeigt die Abbildung 3.2.

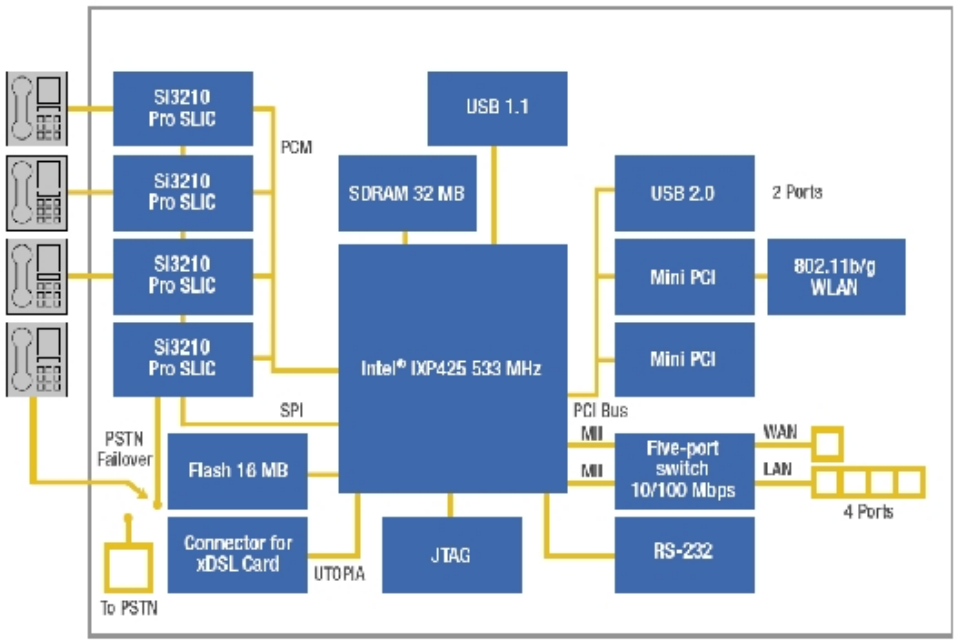

Abbildung 3.2: Aufbau des IXDPG425 (aus [Int04b])

Zur Erweiterung des Boards um eine Wireless-LAN Schnittstelle wurde eine auf dem Atheros 5212-Chipsatz basierende miniPCI-Steckkarte verwendet, welche sowohl die im 2,4 GHz-Band operierenden IEEE 802.11b/g Standards als auch die im 5 GHz-Bereich angesiedelte Norm IEEE 802.11a unterstützt.

Zur Nutzung der speziellen Funktionalität der NPEs ist die Softwarekomponente „Intel Access Library“ notwendig. Diese aktiviert und konfiguriert die jeweilige NPE gemäß der gewünschten Funktion und stellt eine Schnittstelle zur Anwendungsprogrammierung - Application Programming Interface (API) - zur Verfügung. Eine nähere Betrachtung der Programmierschnittstellen des IXP425 erfolgt im Kapitel 5.3.3

Neben der Access Library bietet Intel kaum weitere Software für diese Hardwareplattform. Nur eine speziell auf den IXP425 angepasste Version des Bootloaders Redboot kann von Intel bezogen werden, für ein vollständiges Betriebssystem verweist Intel jedoch auf frei verfügbare sowie kommerzielle Produkte. Dabei wird der IXDPG425 sowohl von Linux als auch von VxWorks[2] und Windows CE .NET unterstützt.

Eine komfortable Möglichkeit zum Entwickeln einer Linux-basierten Anwendung für den IXDGP425 bieten die beiden kommerziellen Produkte Jungo OpenRG[3] sowie MontaVista Linux[4]. Diese können jedoch aufgrund der anfallenden Lizenzkosten nicht für diese Diplomarbeit genutzt werden.

Daher wurde als Entwicklungsumgebung die frei verfügbare Software OpenEmbedded ausgewählt, die im folgenden Kapitel vorgestellt wird.

[2] VxWorks ist ein Echtzeitbetriebssystem, das von Wind River Systems entwickelt wird.
[3] Vgl. Produktbeschreibung OpenRG [Jun05]
[4] Vgl. Produktbeschreibung MontaVista Linux Professional Edition [Mon05]

Kapitel 4

Die Entwicklungsumgebung OpenEmbedded

Zum Vereinfachen der Entwicklung einer auf den Intel IXDPG425 angepassten Linux-Distribution wurde die Open Source-Software OpenEmbedded ausgewählt.

Die Entwicklung von OpenEmbedded begann im Jahre 2002 mit der Portierung einer herstellerunabhängigen Linux-Distribution für einen Personal Digital Assistant (PDA) der Marke Sharp. Das damals noch OpenZaurus genannte Projekt wurde sehr schnell um weitere Pakete und Gerätebeschreibungen erweitert und entwickelte sich daher zu einem Werkzeug für das Portieren unterschiedlichster Anwendungen auf verschiedene „Eingebettete Geräte“.

4.1 Aufbau von OpenEmbedded

Nachdem die bisherige Verzahnung von Programmen zum Portieren und Informationen zum Portier-Vorgang zu einer unübersichtlichen Struktur der OpenEmbedded-Umgebung führte, wurde zum Anfang des Jahres 2005 eine Aufgliederung des Programms in zwei Komponenten vorgenommen:

1. „Bitbake Build System“
2. „OpenEmbedded Metadaten“

Bitbake bezeichnet ein Programm, das verschiedene Aufgaben mit Hilfe der von OpenEmbedded mitgelieferten Metadaten ausführt. Es enthält einen Parser, um Metadaten einzulesen, generiert daraus Abhängigkeitsgraphen, um Paketabhängigkeiten und Aufgabenabhängigkeiten berücksichtigen zu können und erstellt daraus Skripte, welche die Pakete angepasst auf die Zielplattform kompilieren.

Die für den Prozess des Portierens notwendigen Informationen werden im OpenEmbedded Repository, einer Sammlung verschiedenster Metadaten, bereitgehalten.

Das Repository umfasst folgende Bestandteile:

- Klassenbibliotheken (Build Classes)
- Gerätekonfigurationen (Machine Configurations)
- Distributionsbeschreibungen (Distribution Policies)
- Paketbeschreibungen (Recipes)

Die Klassenbibliotheken definieren dabei die für das Portieren eines Paketes notwendigen Aufgaben (Tasks). Die wichtigsten Tasks sind in der folgenden Tabelle dargestellt:

Task	**Beschreibung**
Fetch	Herunterladen von Daten
Unpack	Entpacken eines Archivs
Patch	Anwenden von Patches auf Quellcode
Configure	Durchführen einer Konfiguration des Quellcodes
Compile	Kompilieren des Quellcodes
Stage	Installieren des Paketes in den „Staging“-Bereich
Install	Installieren des Paketes in das Dateisystem
Package	Erstellen eines Paketes

Tabelle 4.1: Tasks in OpenEmbedded (nach [Lau05])

4.2 Konfiguration von OpenEmbedded

Bei dem Vorgang des Portierens eines Paketes werden nun alle bzw. eine Teilmenge der genannten Tasks durchlaufen. Die dafür notwendigen Informationen befinden sich in den Paketbeschreibungen (Recipes), die in den Verzeichnissen unterhalb ./packages zu finden sind. Zum Zeitpunkt des Verfassens dieser Arbeit existierten bereits über 2100 Paketbeschreibungen, aus welchen sich mehr als 5000 Pakete generieren lassen.

Eine einfache Paketbeschreibung lässt sich wie folgt darstellen:

```
DESCRIPTION = "Joe's own editor."
SECTION = "console/utils"
MAINTAINER = "Michael 'Mickey' Lauer <mickey@Vanille.de>"
LICENSE = "GPL"
SRC_URI = "${SOURCEFORGE_MIRROR}/joe-editor/joe-${PV}.tar.gz"

inherit autotools
```

In diesem „Recipe“ sind die folgenden Informationen dargestellt:

- Beschreibung des Paketinhaltes (DESCRIPTION)
- Zuordnung des Paketes (SECTION)
- Herausgeber der Paketbeschreibung (MAINTAINER)
- Lizenz (LICENSE)
- Pfad zum Herunterladen des Paketes (SRC_URI)
- Festlegung der Klasse zum Bearbeiten des Paketes (inherit autotools)

Wenn nun das Programm Bitbake auf dieses Paket angewendet wird, erstellt es mit Hilfe der Klassenbibliothek „autotools“ Skripte, die das Paket „joe“ von dem angegebenen Pfad herunterladen, entpacken, kompilieren und in das Zieldateisystem einfügen.

Erweitert werden kann diese Paketbeschreibung durch die Einbindung von Patch-Dateien zur Manipulation des Quellcodes, sowie der Definition von Abhängigkeiten durch die Variable „DEPENDS“.

Der Compiler benötigt zum Erstellen der Binärdateien jedoch genaue Informationen über das Zielsystem, beispielsweise über die Art der verwendeten Byte-Adressierung. Um diese gerätespezifischen Details von den Paketbeschreibungen (die nicht gerätespezifisch sind) zu entkoppeln, werden solche Informationen in der Gerätekonfiguration vorgenommen.

Im OpenEmbedded Repository existieren bereits eine Vielzahl von Gerätekonfigurationen, abgelegt im Verzeichnis ./conf/machine. Für das gewünschte Zielsystem, den IXDPG425, gab es bisher keine Konfigurationsdatei, jedoch existiert mit dem Projekt OpenSlug[1] eine Konfiguration, welche ebenfalls auf dem Netzwerkprozessor IXP425 basiert und als Grundlage für die Konfiguration des IXDPG425 verwendet werden konnte.

Die Gerätekonfiguration definiert über Variablen die folgenden Parameter:

- Zielarchitektur (TARGET_ARCH)
- Kernelauswahl (PREFERRED_PROVIDER_virtual/kernel)
- Abhängigkeiten (BOOTSTRAP_EXTRA_DEPENDS)
- Compilerparameter (TARGET_CC_ARCH)
- Größe des Dateisystems (IMAGE_ROOTFS_SIZE)

Zusätzlich zu der Gerätekonfiguration lassen sich distributionsspezifische Variablen in den Distributionsbeschreibungen hinterlegen. Diese befinden sich im Verzeichnis ./conf/distro.

[1] OpenSlug ist eine Linux-Distribution für den Linksys NLSU2, einem Gerät zur Kopplung von USB-Festplatten an ein TCP/IP-Netz.

Die folgenden Variablen sind hierbei zu definieren:

- Name der Distribution (DISTRO_NAME)
- Versionsnummer (DISTRO_VERSION)
- Entwicklungsstadium (DISTRO_TYPE)
- Art der Gleitkommarechnung im Zielsystem (TARGET_FPU)
- Zielbetriebsystem (TARGET_OS)
- Art des zu erstellenden Dateisystems (IMAGE_FSTYPES)
- Parameter zum Erstellen des Dateisystems (EXTRA_IMAGECMD)
- Auswahl bevorzugter Pakete (PREFERRED_PROVIDERS)
- Auswahl bevorzugter Versionen von Paketen (PREFERRED_VERSION)

Den Abschluss der Konfiguration von OpenEmbedded bildet die Datei local.conf, die dem Bitbake-System mitteilt, welche Gerätekonfiguration (MACHINE) und welche Distributionsbeschreibung (DISTRO) für die Paketerstellung verwendet werden soll.

Nach der zugegebenermaßen langwierigen Einarbeitung und der aufwendigen Anpassung von OpenEmbedded auf die jeweilige Zielplattform lassen sich nun mit einem einfachen Befehl (bitbake paketname) sämtliche im Repository vorhanden Pakete separat für den IXDPG425 kompilieren. Um eine Vielzahl von Paketen zu einer Distribution zusammenzufassen, können im Verzeichnis „./packages/meta“ Meta-Pakete definiert werden, die dann beim Aufruf mit Bitbake alle benötigten Pakete kompilieren und zu einem auf das Zielsystem passenden Dateisystem zusammenpacken.

Die in dieser Arbeit für die Zielplattform IXDPG425 erstellten Konfigurationsdateien sind auf der beiliegenden DVD, beschrieben im Anhang B, enthalten.

Eine Grundlage für die Auswahl geeigneter Pakete sowie die Anpassung der Paketbeschreibungen stellt die Konzeption der Wireless LAN Distribution dar, die im folgenden Kapitel vorgestellt wird.

Kapitel 5

Konzeption einer Linux Distribution für einen WLAN Access Point

Um den Softwarebedarf für eine Linux-Distribution zum Aufbau eines sicheren Wireless LAN Access Points abschätzen zu können, wurden zuerst kommerzielle WLAN Access Points und Router bezüglich ihrer Funktionalität und Sicherheitseigenschaften untersucht. Daraus resultierend wurden unterschiedliche Einsatzszenarien für den IXDPG425 entwickelt und eine Zuordnung der einzelnen Funktionen des Access Points zu bereits existierenden Open Source Software-Komponenten vorgenommen.

5.1 Analyse kommerzieller WLAN Access Points

Zur Eingrenzung notwendiger Funktionen eines WLAN Access Points für Unternehmensumgebungen wurden unterschiedliche Produkte sowohl vom Marktführer für Netzwerkkomponenten, der Firma Cisco, als auch von dem deutschen Hersteller Funkwerk Enterprise Communications untersucht.

5.1.1 Die Cisco-Aironet Serie

Die Firma Cisco, seit Jahren Marktführer im Bereich IP-Routing und Netzwerktechnologie, deckt mit ihrer Wireless LAN-Technologie „Cisco Aironet“ sämtliche Anwendungsbereiche sowohl kleiner und mittlerer Unternehmen (KMU) als auch die Netzlösungen in Großkonzernen ab. Aus der Größe des Unternehmens und der gewünschten WLAN-Netzabdeckung resultieren unterschiedliche Anforderungen an die Wireless LAN-Technologie: während bei KMU geringe Investitionen und eine unkomplizierte Installation im Vordergrund stehen, sind die Prioritäten von Großkonzernen eher im Bereich Sicherheit und kostengünstige, daher häufig zentrale Wartung angesiedelt.

Aus diesem Grund unterscheidet die Firma Cisco zwei unterschiedliche WLAN-Architekturen:

1. Herkömmliche Access Points mit dezentraler Intelligenz („Thick AP“)
   –> Cisco Distributed WLAN Solution
2. Das Konzept des WLAN-Switching („Thin AP“)
   –> Cisco Centralized WLAN Solution

Im erstgenannten Architekturkonzept sind neben der reinen Funkübertragung viele weitere Funktionen, beispielsweise Sicherheit und Kryptographie, Authentifizierung und Abrechnung, Netzwerkrouting sowie die Vergabe von dynamischen IP-Adressen, direkt auf jedem AP realisiert. Daher muss jeder AP sowohl vor Inbetriebnahme als auch im laufenden Betrieb konfiguriert und ständig überwacht werden. Während in einem kleinen Unternehmen diese Aufgabe sicherlich einfach zu realisieren ist, kann dies bei einer Großinstallation von mehreren hunderten Access Points zu erheblichen Problemen vor allem im Bereich der Sicherheit führen.

Dieses Problem hat Cisco erkannt und mit der Übernahme der Firma Airespace das Konzept des WLAN-Switching eingeführt. Hierbei kommt dem Access Point nur noch die Aufgabe der Funkübertragung zu; alle weiteren Funktionen werden durch einen zentralen, um die für Wireless-LAN notwendigen Funktionen erweiterten Switch durchgeführt. Aufgrund des geringeren Hardwareaufwandes im AP ergeben sich dadurch geringere Kosten bei der Anschaffung einer großen Menge von Access Points, außerdem verhilft dieses Konzept zu einer einfachen, zentralen Konfiguration und Wartung.

Zu den typischen Vertretern des ersten Architekturkonzeptes gehört der Access Point „Cisco Aironet 1300“. Er beinhaltet die folgenden Funktionen:

- IEEE 802.11a/g Funkschnittstelle
- WEP, WPA, WPA2 Verschlüsselung
- WEP und WPA-PSK Authentifizierung
- Authentifizierung auf Basis von IEEE 802.1X (externer RADIUS-Server notwendig)
- Unterstützung von Netzwerkprioritäten (Quality of Service (QoS))
- Konfiguration per Web, Telnet und Simple Network Management Protocol (SNMP)

Jeder AP im Unternehmen muss vollständig konfiguriert werden und somit beispielsweise Sicherheitsparameter, Kanaleinstellungen und Service Set Identifier (SSID) erhalten. In größeren Installationen lässt sich dies durch den Einsatz einer Management-Software vereinfachen. Bei der Verwendung von IEEE 802.1X wird zusätzlich ein Authentifizierungs-Server benötigt, den Cisco unter dem Namen Cisco Secure Access Control Server vertreibt.

Im zweiten Architekturkonzept wird der Access Point von sämtlichen Aufgaben außerhalb der Funkübertragung entlastet - daher wird der AP dann auch als Cisco Aironet 1000 Series Lightweight Access Point bezeichnet. Dieser „leichtgewichtige“ AP ist mit einem WLAN Controller verbunden, der zentral für eine Vielzahl von Access Points die Konfiguration, das

Routing und Roaming sowie die Sicherheitsfunktionen übernimmt. Auch hierbei werden sämtliche aktuellen Sicherheitsarchitekturen wie WEP, WPA und WPA2/IEEE 802.11i unterstützt. Weitergehende Sicherheitsmechanismen wie Virtuelle Private Netzwerke werden durch andere Produktlinien bereitgestellt.

Als Ergebnis der Analyse der Cisco WLAN-Produkte lässt sich feststellen, dass die softwaremäßige Ausstattung eines Access Point stark vom Einsatzbereich und Kundenkreis abhängig ist. Ein kleines Unternehmen, das sein bisheriges LAN um eine drahtlose Schnittstelle in einigen wenigen Räumen erweitern möchte und bisher keine Netzwerk-Management-Lösung einsetzt, wird wohl einen komplett ausgestatteten AP mit sämtlichen Funktionen „out of the box“ bevorzugen. Im Gegensatz dazu wird bei einer großen WLAN-Installation die Switching-Architektur kostengünstiger und vom Administrationsaufwand einfacher einzuführen sein.

Einen anderen Weg verfolgt die Firma Funkwerk Enterprise Communications mit Sitz in Nürnberg, die unter der Bezeichnung „bintec X2250“ einen sehr reichhaltig ausgestatteten Access Point für kleine und mittelständige Unternehmen anbietet.

5.1.2 Der Bintec X2250

Als Netzwerkprodukt mit umfangreichen Routing-Funktionalitäten und einem abgesichterten kabellosen Netzzugang wird der Bintec X2250 angeboten[1]. Diese Vorgaben werden angesichts des enormen Funktionsumfanges überzeugend eingehalten.

Der für ca. 400 Euro erhältliche WLAN-Router umfasst die folgenden Funktionen:

- 2 Ethernet-Schnittstellen
- 1 ISDN-Schnittstelle
- IEEE 802.11a/g Funkschnittstelle
- Routing mit Quality of Service (QoS) und Optimierung für Voice over IP (VoIP)
- Firewall mit Stateful Packet Inspection
- WLAN-Absicherung mit WEP, WPA und WPA2, 802.1X mit RADIUS-Authentifikation über externen RADIUS-Server
- Multi-SSID für unterschiedliche Benutzergruppen im WLAN
- Wireless Distribution System (WDS) zur drahtlosen Kopplung mehrerer Access Points
- Integrierter IPSec-Server und -Client mit Unterstützung von X.509-Zertifikaten (Standardlizenz: 5 Tunnelverbindungen)
- Unterstützung von PPP over Ethernet (PPPoE) zur DSL-Einwahl
- Konfiguration per Web-Formular oder Secure Shell (SSH)-Verbindung

[1] Vgl. Produktbeschreibung Bintec X2250 [Fun05]

Mit diesem Router ist es möglich, ein kleines Unternehmen über eine DSL-Leitung sicher an das Internet zu koppeln, einen drahtlosen Zugang in das Internet und in das LAN zu ermöglichen und diesen entweder mit dem aktuellsten Sicherheitsstand IEEE 802.11i oder mit IPSec abzusichern. Zusätzlich können sich Mitarbeiter über das Internet per IPSec mit dem Router verbinden und damit einen sicheren Zugang zum Firmennetzwerk erlangen.

Das Problem unterschiedlicher Benutzergruppen im Wireless LAN und die damit resultierenden Schwierigkeiten der Rechtevergabe löst der Router über die Einführung der neuen Multi-SSID-Technologie, bei der ein Wireless Interface mehrere SSID annehmen kann und jeder SSID eine eigene Sicherheitsrichtlinie zugeordnet wird. So kann beispielsweise der Zugang zum internen LAN über eine WPA-Radius Authentisierung erfolgen, während das Internet bereits durch die Konfiguration eines WEP-Schlüssels genutzt werden kann.

Selbst die für die WPA- und IPSec - Authentisierung notwendigen Zertifikate (X.509) lassen sich über eine Konfigurationsschnittstelle auf dem Router erstellen und verwalten.

5.1.3 Zusammenfassung

Beim Vergleich der Funktionalitäten der Produkte von Cisco und Bintec/Funkwerk zeigt sich ein großer Unterschied in der Produktphilosophie beider Unternehmen: während Cisco seine WLAN Access Points grundlegend spartanisch ausstattet und sämtliche Zusatzfunktionen über andere Produktlinien (Management-Software, VPN-Router, Switches etc.) anbietet, verfügt der Bintec-Router über eine Ausstattungsvielfalt, die kaum noch zu überblicken ist.

Zwar verfügt der Bintec-Router über eine Schnittstelle zu dem Programm XADMIN, mit dem sich mehrere AP's gleichzeitig konfigurieren lassen - eine durchgehende Integration in eine Netzwerk-Management-Lösung ist jedoch nicht vorgesehen.

Somit stellt sich ein reichhaltig ausgestatteter Access Point in einer kleinen Unternehmensumgebung als sehr vorteilhaft heraus, da keine weiteren Software- oder Hardwarekomponenten zum Betrieb benötigt werden. In größeren Installationen versagt dieses Konzept jedoch aufgrund der mangelnden Skalierbarkeit.

Basierend auf den bisherigen Analysen sollen nun einige Einsatzszenarien für einen auf Open Source Software basierenden sicheren WLAN-Access Point vorgestellt werden.

5.2 Einsatzszenarien für WLAN Access Points

Die umfangreiche technische Ausstattung des IXDPG425 sowie die große Vielfältigkeit von Open Source Software erlaubt eine Vielzahl von möglichen Einsatzszenarien des WLAN Access Point bzw. Router.

In diesem Kapitel sollen nun verschiedene Szenarien vorgestellt werden, in dessen Folge dann Open Source Softwarekomponenten zur Erfüllung dieser Funktionen ausgesucht werden.

5.2.1 WLAN Access Point ohne Sicherheitsfunktionen

Im einfachsten Fall erfolgt durch den Access Point, wie in der folgenden Abbildung gezeigt, ausschließlich die Herstellung einer Funkschnittstelle.

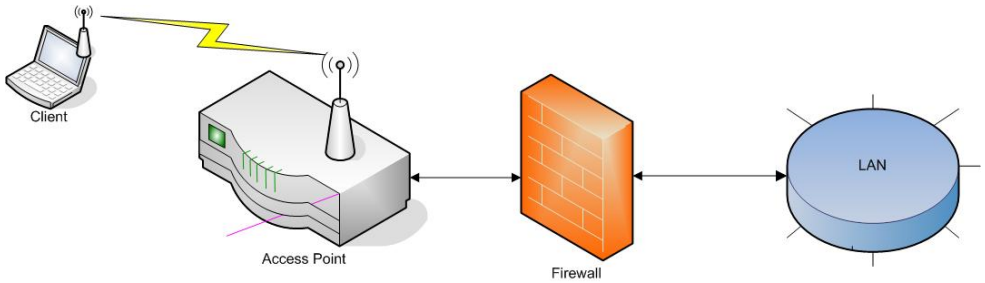

Abbildung 5.1: WLAN Access Point ohne Sicherheitsfunktionen

Die drahtlose Schnittstelle wird dabei über eine softwareseitige Bridge mit dem Ethernet-Interface des Routers verbunden. Somit werden sämtliche Pakete von per Funk verbundenen Clients an die Firewall weitergeleitet, erst dort erfolgt das Verwerfen von Nachrichten unberechtigter Absender. In der Firewall könnte weiterhin auch ein VPN-Server implementiert sein, der dann eine Verschlüsselung der WLAN-Kommunikation ermöglicht.

Dieses Szenario ist beim Vorhandensein eines VPN-Servers im Unternehmen durchaus denkbar. Die Einfachheit der Konfiguration und die Nutzung bestehender Ressourcen ermöglicht somit eine schnelle Einbindung des WLAN in das Unternehmens-Netzwerk. Mit einem Regelwerk in der Firewall kann dafür gesorgt werden, dass nur Clients, die eine autorisierte IPSec-Verbindung nutzen, Zugriff auf das Unternehmens-Netzwerk erhalten.

Wenn kein IPSec-Server vorhanden ist, kann der Access Point ebenfalls um diesen erweitert werden.

5.2.2 WLAN Access Point mit IPSec-Sicherheit

In Unternehmen, in denen eine Anwendung der WLAN-Sicherheitsarchitekturen aus Kompatibilitätsgründen nicht erwünscht ist und bisher auch keine VPN-Infrastruktur vorhanden ist, muss diese Funktionalität vollständig durch den um IPSec erweiterten Access Point erbracht werden.

Wie in Abbildung 5.2 gezeigt, befindet sich nun im Access Point sowohl die Firewall als auch der IPSec-Server. Wenn ein Client eine drahtlose Verbindung ins LAN aufbauen will, muss er einen IPSec-Client benutzen und zu Beginn der Authentisierung einen gültigen Schlüssel bzw. ein gültiges Zertifikat vorweisen. Die Firewall im Access Point ist dabei so konfiguriert, dass nur Pakete, die mit IPSec den Access Point erreicht haben, in das LAN-Netz geroutet werden dürfen. Netzwerk-Pakete, welche über eine ungesicherte Verbindung die drahtlose Schnittstelle erreichen, werden sofort verworfen.

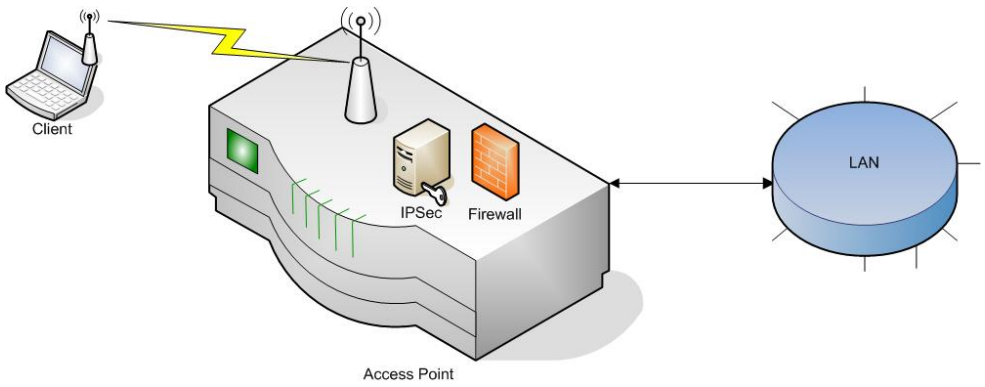

Abbildung 5.2: WLAN Access Point mit IPSec

Somit kann zum einen die Vertraulichkeit der über das WLAN übertragenen Daten sichergestellt werden, zum anderen wird dadurch eine sichere Zugriffskontrolle auf das LAN ermöglicht.

Um jedoch unberechtigte Nutzer schon auf Seite der WLAN-Kommunikation den Zugang zum Netzwerk zu sperren, bedarf es der Anwendung der Sicherheitsarchitektur IEEE 802.11i.

5.2.3 WLAN Access Point mit IEEE 802.11i

Für die Absicherung von Wireless LAN-Systemen auf Ebene der drahtlosen Schnittstelle steht neben der WEP-Sicherheitsarchitektur nun auch die WPA/WPA2-Architektur der Norm 802.11i zur Verfügung. Bereits im ersten Kapitel dieser Arbeit wurde gezeigt, dass die Sicherheit von WEP für Unternehmensanwendungen völlig unzureichend ist, weshalb auf eine weitere Betrachtung dieser verzichtet werden soll.

Stattdessen soll der Einsatz der bisher als sicher geltenden WPA/WPA2-Architektur beschrieben werden. Da sich der Einsatz von WPA-PSK, also der Nutzung eines gemeinsamen Schlüssels, nur im Privateinsatz empfiehlt, ist die Einführung eines Authentifikations-Servers unumgänglich. Die Abbildung 5.3 zeigt das Szenario eines WLAN AP mit WPA-Enterprise.

Beim Aufbau einer WLAN-Verbindung muss sich nun der Client authentifizieren. Dies geschieht beispielsweise durch den Versand eines gültigen Zertifikates an den RADIUS-Server. Nach erfolgreicher Authentifizierung transferiert der RADIUS-Server den Sitzungsschlüssel an Client und Access Point und bildet somit die Grundlage für eine sichere Kommunikation über die drahtlose Schnittstelle.

Der dafür notwendige RADIUS-Server kann sich entweder im Unternehmensnetzwerk (LAN) oder im Access Point befinden. Bei Vorhandensein eines solchen Servers ist seine Nutzung angebracht. In Unternehmen, die nicht über einen RADIUS-Server verfügen, könnte diese Aufgabe auch durch eine Software im Access Point übernommen werden (unter Voraussetzung ausreichender Ressourcen).

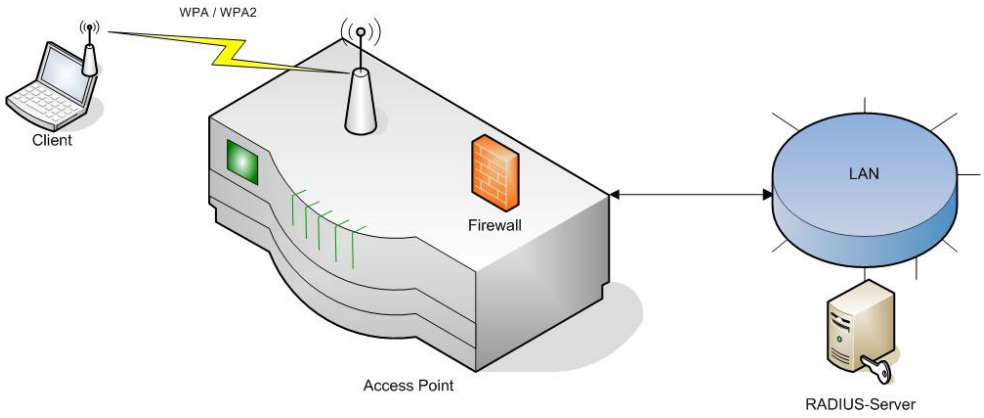

Abbildung 5.3: WLAN Access Point mit WPA-Enterprise

Die Firewall lässt sich nun ebenfalls direkt im Access Point realisieren, da aufgrund der WPA-Authentifizierung nur noch berechtigte Nutzer Zugriff auf den AP bekommen. In der Firewall muss dann noch geregelt werden, welche Dienste authentifizierte Clients benutzen dürfen und wie der Acess Point vor unberechtigter Manipulation geschützt werden kann.

Ob die Verschlüsselung mit WPA oder WPA2 durchgeführt wird, ist von der gewünschten Client-Kompatibilität abhängig. Während in der WPA-Architektur das TKIP-Protokoll zur Verschlüsselung der Daten verwendet wird, nutzt WPA2 die als CCMP bezeichnete Variante des wesentlich sichereren AES-Algorithmus. Jedoch unterstützen noch nicht alle WLAN-Adapter diesen neuen Standard, weshalb auf absehbare Zeit zur Wahrung der Kompatibilität noch die ältere WPA-Norm verwendet werden sollte.

Auch in diesem Szenario wird aus Vereinfachungsgründen die drahtlose Schnittstelle mit der Ethernet-Schnittstelle über eine Bridge verbunden. Notwendige und weitere optionale Dienste wie beispielsweise DHCP zur IP-Adressvergabe sowie ein Zugang ins Internet müssen über Server im LAN bereitgestellt werden.

Wenn diese allgemeinen Netzwerkdienste auch noch in das Szenario integriert werden sollen, verlässt man die Produktklasse „Access Point“ und konstruiert einen WLAN Router mit deutlich erweitertem Funktionsumfang.

5.2.4 WLAN Router

Ergänzt man den bisherigen Access Point um die Möglichkeit des Routings in Fremdnetzwerke (zumeist das Internet), so entsteht ein WLAN Router. Neben der Einwahl in das Internet sollte dieser weitere Netzwerkdienste zur Vergabe von IP-Adressen, zur Weiterleitung von Domain Name System (DNS)-Anfragen und zur Absicherung des Routers vor unberechtigten Manipulationen zur Verfügung stellen.

Somit kann ein solcher in der folgenden Abbildung gezeigter WLAN Router sämtliche für ein kleines Unternehmen notwendigen Netzwerkdienste anbieten.

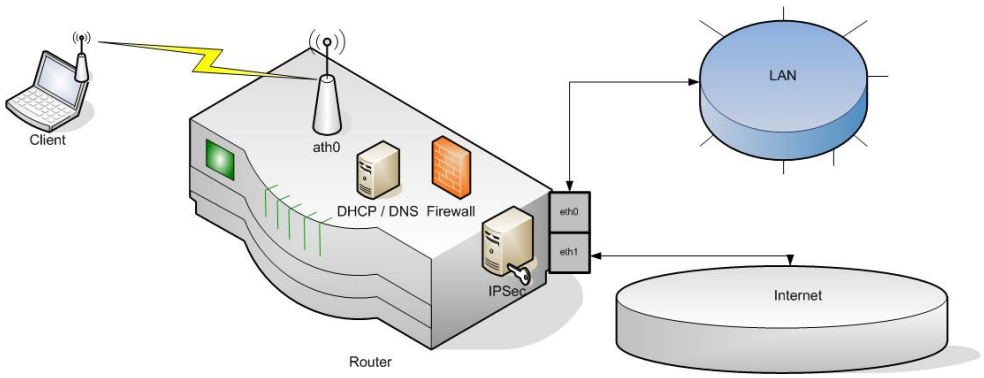

Abbildung 5.4: WLAN Router

Der WLAN Router nutzt alle drei Netzwerk-Interfaces, die der IXDPG425 zur Verfügung stellt. Dabei können sich drahtlose Computer über die WLAN-Schnittstelle am Router anmelden. Die Authentifizierung wird entweder mit IPSec oder mit WPA/WPA2 durchgeführt. Über das erste Ethernet-Interface kann das Unternehmensnetzwerk mit dem Router verbunden werden, um eine Nutzung der über den Router angebotenen Dienste zu ermöglichen und wechselseitigen Zugriff zwischen drahtlosen und im LAN befindlichen PCs zu gewährleisten. Das zweite Ethernet-Interface verbindet den Router mit dem Internet. Dies kann beispielsweise über ein DSL-Modem erfolgen; hierfür ist jedoch die Nutzung des PPPoE-Protokolls notwendig.

Zur Vereinfachung der Client-Konfiguration sollten außerdem elementare Netzwerkdienste durch den Router implementiert werden. So kann die Vergabe von IP-Adressen und Routinginformationen automatisch über einen Dynamic Host Configuration Protocol (DHCP)-Server durchgeführt werden, das für die Nutzung des Internets notwendige Weiterleiten von DNS-Anfragen übernimmt ein DNS-Forwarder auf dem Router.

Wenn beispielsweise bei der DSL-Einwahl nur eine externe IP-Adresse zur Kommunikation mit dem Internet vorhanden ist, müssen sämtliche Internet-Anfragen aller Clients mit dieser externen IP-Adresse in das Internet gesendet werden. Diese Funktionalität wird durch Network Address Translation (NAT) in der Firewall zur Verfügung gestellt. Zudem sorgt die Firewall für das Blockieren unberechtigter Anfragen aus dem Internet sowie den Schutz des Routers vor unautorisierter Manipulation durch Benutzer im WLAN oder LAN.

Die Umsetzung dieses Szenarios auf dem IXDPG425 würde einen komplexen WLAN Router entstehen lassen, der einem kleinen Privat- oder Unternehmensnetzwerk ohne weitere Hard- bzw. Softwarekomponenten sowohl den gemeinsamen Internet-Zugang ermöglicht als auch einen sicheren und komfortablen Wireless LAN Zugriff gewährleistet.

Wünschenswert wäre jedoch noch eine zusätzliche Differenzierung unterschiedlicher Benutzergruppen beim Zugriff auf die WLAN-Schnittstelle. Das folgende Szenario soll die Problemstellung sowie verschiedene Lösungsmöglichkeiten vorstellen.

5.2.5 WLAN Router mit unterschiedlichen Benutzergruppen

Die bisher vorgestellten Methoden zur WLAN-Authentifizierung erfordern zumindest das Vorhandensein des Zertifikates der „vertrauenswürdigen Zertifizierungsstelle“ auf dem Client-PC. Bei Verwendung von IPSec bzw. WPA mit EAP/TLS wird außerdem das persönliche Zertifikat des Benutzers benötigt. Somit ist es schwierig, bzw. unmöglich, einem Gast kurzfristig Zugang zum abgesicherten Wireless LAN zu ermöglichen. Selbst wenn diese organisatorische Hürde durch Ausstellen und Installation eines temporären Zertifikates überwunden wird, hat das System keine Möglichkeit festzustellen, ob der Zugang von einem Mitarbeiter oder einem Gast verwendet wird. Damit kann es auch keine auf den Anwender angepasste Sicherheitsregel anwenden. Wünschenswert wäre hier beispielsweise eine Sicherheitsarchitektur, die es Mitarbeitern mit drahtlosen Endgeräten erlaubt, sowohl das Internet als auch das Unternehmensnetz (LAN) zu erreichen, Gästen jedoch den Zugriff auf das LAN verbietet.

Diese Problematik hat auch die WLAN-Industrie erkannt und das Konzept der Multi-SSID Geräte vorgestellt. Ein Access Point mit einem Funkinterface sendet auf einem Kanal mehrere unterschiedliche Funkkennungen (SSID) und kann dabei für jede Funkkennung eine andere Sicherheitsregel vorgeben.

Somit können für Mitarbeiter als auch für Gäste folgende Zugriffe ermöglicht werden:

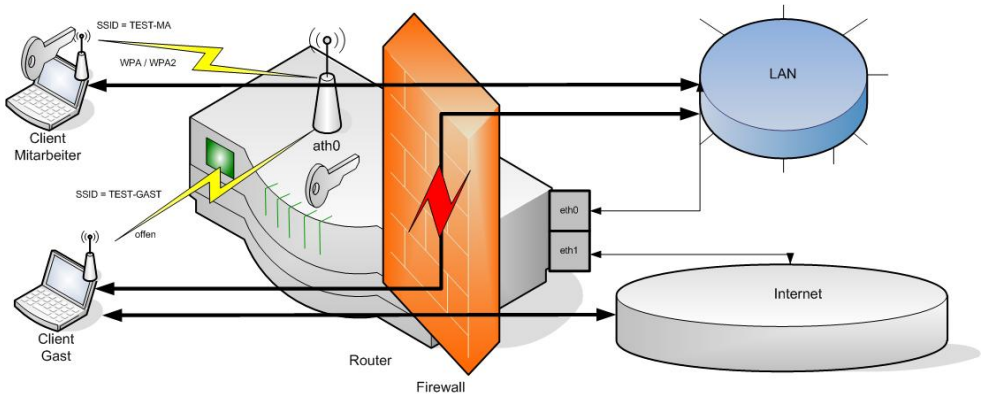

Abbildung 5.5: WLAN Router mit Multi-SSID

In dem abgebildeten Szenario ist sowohl der Client Mitarbeiter als auch der Client Gast mit dem Router über eine WLAN Assoziation verbunden. Der Mitarbeiter hat sich jedoch mit der SSID „TEST-MA“ angemeldet und musste zur Authentisierung sein Zertifikat vorweisen. Der Gast nutzte die SSID „TEST-GAST“ und konnte sich somit ohne Authentifizierung anmelden. Die Firewall kann somit die Herkunft der Pakete bestimmen und erlaubt Zugriffe auf das LAN nur aus der SSID „TEST-MA“. Versucht der Gast das Versenden eines Paketes in das LAN, wird dieses von der Firewall blockiert, Zugriffe in das Internet werden allerdings ermöglicht.

Die Multi-SSID Technologie, die Voraussetzung für dieses Szenario ist, existiert jedoch nicht in den WLAN-Standards der IEEE, sondern stellt eine herstellerspezifische Erweiterung dar, die bisher nur in einigen wenigen Produkten verfügbar ist.

Bei Nichtverfügbarkeit von Multi-SSID könnte eine mögliche Lösung der Problemstellung folgendermaßen realisiert werden: Es muss ein Weg gefunden werden, um eine Zuordnung von MAC-Adresse des Clients auf den Benutzernamen bzw. die Benutzergruppe desselben zu ermöglichen. Bei der Verwendung von WPA/WPA2 könnte hierbei der Anmeldename herangezogen werden. Bei Kenntnis des Benutzernamens und einer Überprüfung der entsprechenden Berechtigungen könnte nun skriptgesteuert eine Anpassung der Firewallregeln für diese MAC-Adresse durchgeführt und somit beispielsweise der Zugriff auf das LAN erlaubt werden.

Hierbei muss allerdings sichergestellt werden, dass bei jeder WLAN - Assoziierung und Deassoziierung dieser Adressfilter aktualisiert wird. Nur so lässt sich vermeiden, dass ein Gast die MAC-Adresse eines authentifizierten Mitarbeiters nutzt, um unberechtigt Zugriff auf bestimmte Bereiche des Netzwerkes zu erlangen.

5.3 Realisierung eines WLAN Access Points mit Open Source Software

Um das Ziel der Realisierung der im vorherigen Kapitel beschrieben Szenarien für einen Enterprise WLAN Access Point erreichen zu können, ist die Integration unterschiedlicher Open Source Software-Komponenten zu einer auf dem IXDPG425 lauffähigen Linux Distribution notwendig.

Daher sollen in den folgenden Kapiteln verschiedene Open Source Projekte bezüglich ihrer Eignung der Nutzung in einer Embedded Linux Distribution untersucht werden.

Bei der Auswahl von Software müssen unterschiedliche Kriterien beachtet werden:

Offenheit Besonders in sicherheitskritischen Anwendungen ist die Offenheit des Quellcodes ein entscheidendes, vertrauensbildendes Merkmal. Nur so kann eine Überprüfung des Programms auf Korrektheit und das Nichtvorhandensein unerwünschter Eigenschaften gewährleistet werden. Änderungen am Quellcode, beispielsweise durch Patches, sollten nachvollziehbar sein.

Lizenzen Es soll möglichst nur freie Software, beispielsweise den Lizenzbedingungen der GNU General Public License (GPL) folgend, eingesetzt werden. Bei Abweichungen von diesen Lizenzbedingungen muss die Software zumindest kostenfrei nutzbar sein.

Ressourcensparsamkeit Aufgrund der beschränkten Speicherressourcen sowohl im Flash als auch im RAM des IXDPG425 soll nur Software mit sparsamen Ressourceneinsatz angewendet werden.

Stabilität Zur Sicherung der Funktionsfähigkeit soll ausschließlich getestete und nicht noch im Entwicklungsstadium befindliche Software verwendet werden. Bei der Verwendung von „unstable“-Versionen (Entwicklungsversionen) müssen eventuelle Einschränkungen und Fehler dokumentiert sowie Alternativen dargestellt werden.

Erweiterbarkeit Bei der Auswahl von Software soll möglichst darauf geachtet werden, dass das Gesamtsystem modular und erweiterbar aufgebaut ist. Nur somit kann eine Anpassung an zukünftige Technologien gewährleistet werden.

Auf Basis dieser Anforderungskriterien wurden nun verschiedene Open Source Projekte bezüglich ihrer Eignung für eine auf dem IXDPG425 lauffähige WLAN Distribution untersucht. Die Ergebnisse dieser Evaluation werden in den folgenden Kapiteln dargestellt.

5.3.1 Erstellung einer Toolchain

Eine geeignete Toolchain ist die Voraussetzung zum Kompilieren von Software für die Zielplattform IXDPG425 in einer herkömmlichen Linux Umgebung. Die Toolchain ist die eigentliche Cross - Entwicklungsumgebung auf dem Entwicklungsrechner. Sie stellt die nötigen Werkzeuge zum Erstellen des Cross Compilers bereit und dient später der Cross - Softwareentwicklung.

Als Cross Compiling versteht man das Erstellen von Objekt- oder Maschinencode, der nicht kompatibel zum Prozessor des kompilierenden Computers ist, sondern zu einem Prozessor eines anderen Computers.

Die Toolchain besteht aus folgenden Komponenten:

- binutils: Zusammenstellung von Werkzeugen für den Umgang mit Binärdateien der Ziel-Prozessorarchitektur
- gcc: der Cross Compiler
- c-Library: eine speziell in Größe und Funktionsumfang auf das Embedded System angepasste C-Bibliothek

Bei der Auswahl einer geeigneten Toolchain musste vor allem die Kompatibilität zum dem im weiteren Verlauf der Arbeit genutzten Madwifi-Treiber bedacht werden. Dieser Treiber für die WLAN-Schnittstelle besteht aus einem binären und einem quelloffenen Bestandteil. Dabei wird seitens der Entwickler empfohlen, beim Kompilieren der Quellen dieselbe Toolchain zu nutzen, die diese für den Binär-Bestandteil verwendet haben. Um einen Mix aus verschiedenen Toolchains zu vermeiden, wurden daher konsequent die folgenden Programme verwendet:

- binutils: Version 2.14
- gcc: Version 3.3.4
- glibc: Version 2.3.2

OpenEmbedded unterstützt die Erstellung der Toolchain durch vorhandene Paketbeschreibungen sowie einigen Patches zur Optimierung des Compilers für die Xscale-Architektur.

5.3.2 Auswahl eines Kernels und einer Kernelkonfiguration

Eine für den weiteren Verlauf dieser Diplomarbeit kritische Entscheidung stellt die Auswahl eines geeigneten Linux Kernels dar. Noch immer existieren zwei unterschiedlichen Entwicklungsstränge:

1. der in die Jahre gekommene 2.4er Kernel, der als sehr stabil gilt, jedoch nur noch sehr langsam weiterentwickelt wird und daher kaum neue Funktionalität erhält
2. der seit Dezember 2003 freigegebene Kernel der Reihe 2.6, welcher mit einer Vielzahl von Verbesserungen versehen wurde und auf dem das Hauptaugenmerk der Weiterentwicklung liegt

Das für den Netzwerkbereich entscheidende Abgrenzungsmerkmal des neuen 2.6er Kernels ist die konsequente Unterstützung des neuen Internet Protokoll Standards IPv6. Aus diesem Grunde wurde auch das IPSec-Protokoll in den Kernel übernommen und braucht daher nicht mehr als spezielle Software im Benutzerbereich geladen werden.

Vor allem zur Sicherung der Erweiterbarkeit wurde bewusst die Entscheidung getroffen, einen aktuellen Kernel der 2.6er Serie für die Implementation zu verwenden. Dabei fiel die Wahl auf den seit März 2005 in der stabilen Version vorliegenden Kernel 2.6.11.2, mit dem bereits die Programmierer des NLSU2[2] positive Erfahrungen gesammelt haben. Der größte Vorteil dieses Kernels liegt in der vollständigen Unterstützung der verwendeten Hardware. Während bei einem älteren Kernel umfangreiche Quellcode-Änderungen notwendig sind, kann der 2.6er Kernel direkt für den IXDPG425 konfiguriert werden. Außer einem Patch zur Anpassung der Maschinennummer sind keine weiteren Veränderungen zum Kompilieren notwendig.

Bei der Konfiguration des Kernels konnte auch auf eine Konfigurationsdatei aus dem OpenSlug-Projekt zurückgegriffen werden, die sich im OpenEmbedded Repository befindet.

Neben der Definition von gerätespezifischen Parametern für den IXDPG425 wurde die Konfiguration des Kernels um die folgenden Module erweitert:

- IPSec Unterstützung (Netkey)
- Firewall (Netfilter)
- Bridge
- Wireless LAN
- Point-to-Point Protokoll (PPP)
- PPP over Ethernet (PPPoE)
- Universal Seriell Bus (USB)
- SCSI Unterstützung (notwendig für USB Speicheranwendungen)

[2] Network Storage Link von Linksys

- Network File System (NFS)
- Unterstützung kryptographischer Operationen (Crypto)

Die notwendigen Schritte zum Anpassen und Kompilieren des Kernels werden mit OpenEmbedded durchgeführt. Dafür wurde ein neues Paket namens „openixp-kernel-2.6.11.2“ erstellt, das eine funktionsfähige Kernelkonfiguration und den notwendigen Patch enthält.

5.3.3 Die Intel Access Library

Wie bereits bei der Vorstellung der Hardwareplattform erwähnt, ist eine von Intel herausgegebene Software notwendig, um spezielle Funktionen des IXP425 nutzen zu können. Die so genannte „Intel Access Library“ ist quelloffen, jedoch untersteht sie einer speziellen Intel Lizenz und darf daher nur mit auf Intel Netzwerkprozessoren basierenden Geräten vertrieben werden. Aus diesem Grund ist eine direkte Integration dieses Treibers in das Linux Kernel-Paket nicht möglich.

Die Intel Access Library stellt eine Programmierschnittstelle (API) zur Verfügung, mit deren Hilfe vor allem die Funktionen der NPEs genutzt werden können. Daher ist sie notwendig, um die auf den NPEs basierenden Ethernet-Schnittstellen nutzen zu können.

Dabei existieren jeweils zwei unterschiedliche Versionen der Access Library: eine frei herunterladbare Version ohne Unterstützung kryptographischer Operationen und eine exportbeschränkte Version mit Krypto-Unterstützung. Die in den USA geltenden Exportbeschränkungen für kryptographische Erzeugnisse machen diese Aufteilung notwendig - daher ist zum Herunterladen der Access Library mit Krypto-Unterstützung eine vorherige Registrierung und Anfrage an den Intel Export-Koordinator notwendig.

Zur Nutzung der integrierten Ethernet-Schnittstellen ist außerdem noch ein spezieller Ethernet-Treiber notwendig, welcher der GPL untersteht.

OpenEmbedded besitzt im Repository bereits Paketbeschreibungen, sowohl für die Access Library (ixp4xx-csr-1.4) als auch den Ethernet-Treiber (ixp425-eth-1.1). Aufgrund der Exportbeschränkung kann aber nur die Version ohne kryptographische Unterstützung kompiliert werden. Abhilfe schafft hier das manuelle Herunterladen der kryptographischen Komponenten, Einfügen dieser in die Quelltextstruktur der Access Library sowie das manuelle Anpassen der verwendeten Makefiles. Zusätzlich muss auch der Ethernet-Treiber manuell angepasst werden, um in der „Ethernet NPE B“ die Unterstützung der Verschlüsselungsalgorithmen zu aktivieren.

5.3.4 WLAN-Treiber und Wireless-Tools

Die in dieser Arbeit eingesetzten WLAN-Adapter verfügen über einen Chipsatz der Firma Atheros. Für diese Wireless LAN Schnittstelle bietet der aktuelle Kernel bisher noch keine Unterstützung. Jedoch existiert mit dem MADWIFI-Projekt[3] eine Open Source Software, welche eine freie Implementierung des Treibers für diese Adapter zur Verfügung stellt.

Diese Implementierung gliedert sich in 3 Hauptkomponenten:

1. der „Atheros Hardware Access Layer“ (ATH_HAL)
2. die Geräteschnittstelle (ATH_PCI)
3. die 802.11-Schnittstelle (WLAN)

Um den Anforderungen der Federal Communications Commission (FCC)[4] zu genügen, wird die Hardwareschnittstelle Hardware Access Layer (HAL) nur als Binärcode zur Verfügung gestellt. Somit können Veränderungen an Funkparametern, z.B. Erhöhung der Ausgangsleistung und Verwendung unerlaubter Kanäle verhindert werden.

Damit widerspricht dieser Treiber dem Kriterium der Offenheit, jedoch muss diese Einschränkung in Kauf genommen werden, da ansonsten der weit verbreitete Atheros-Chipsatz unter Linux nicht in Betrieb genommen werden kann.

Das „Atheros Hardware Access Layer“ im Binärformat wird bereits in einer für den IXDPG425 geeigneten Version (xscale-be-elf) angeboten. Beim Kompilieren der quelloffenen Module muss nun darauf geachtet werden, möglichst die gleiche Toolchain zu verwenden, die vom Entwickler zum Erstellen des HAL verwendet wurde. Bei Missachtung dieser Vorgabe kann die Funktionsfähigkeit des Treibers nicht hergestellt werden.

Wesentlich größere Probleme bereitet jedoch das frühe Entwicklungsstadium dieser Software, die explizit als „Beta“ gekennzeichnet ist und bisher noch nicht in einer stabilen Version verfügbar ist. Der Einsatz dieses WLAN-Adapters in Produktivsystemen ist deshalb bisher nicht zu empfehlen.

Einen möglichen Ausweg bietet die Verwendung von WLAN-Adaptern eines anderen Herstellers. So existiert für die Chipsätze Prism2/2.5/3 der Firma Intersil eine Open Source Implementation namens HostAP[5], welche statt einer HAL die Firmware zur Kapselung der Funkparameter verwendet und bereits seit einiger Zeit als stabile Version verfügbar ist.

Aufgrund der fehlenden Verfügbarkeit dieser Adapter zur Anfertigung dieser Arbeit müssen die genannten Einschränkungen in Kauf genommen werden.

Neben dem gerätespezifischen WLAN-Treiber müssen im Kernel noch die Erweiterungen für Wireless LAN (Wireless Extensions) aktiviert werden. Dies geschieht durch die Zeile CONFIG_NET_RADIO=y in der Kernelkonfiguration.

[3] Multiband Atheros Driver for WiFi
[4] Zulassungsbehörde für Kommunikationsgeräte in den USA
[5] Vgl. Host AP driver for Intersil Prism2/2.5/3, hostapd, and WPA Supplicant [Mal05a]

Außerdem wird eine Software zur Konfiguration der drahtlosen Schnittstelle benötigt. Hierfür nutzt man das Paket „wireless-tools“, das im OpenEmbedded Repository in der Version 28 vorliegt. Dieses Werkzeug stellt den Befehl „iwconfig“ zur Verfügung, mit welchem ähnlich zu „ifconfig“ spezifische WLAN-Parameter konfiguriert werden können.

5.3.5 Unterstützung kryptographischer Operationen

Sämtliche im weiteren Verlauf dieser Arbeit genutzten Sicherheitsprogramme benötigen Bibliotheken zur Ausführung kryptographischer Operationen. Diese werden durch das Paket „openssl“ zur Verfügung gestellt, das aufgrund des Vorhandenseins im Repository von OpenEmbedded automatisch kompiliert werden kann.

Dabei nutzt OpenSSL jedoch ausschließlich den Prozessorkern zur Ausführung der kryptographischen Operationen. Um von den Vorteilen der Krypto-NPE im Intel IXP425 profitieren zu können, ist die Unterstützung asynchroner Krypto-Operationen notwendig. Das Ziel ist dabei, sämtliche Verschlüsselungsoperationen, die durch die Hardwarebeschleunigung des IXP425 durchgeführt werden können, an die NPE abzugeben.

OpenSSL unterstützt diese asynchronen Krypto-Operationen durch die Einführung des Parameters „engine cryptodev“. Dafür ist jedoch ein Gerätetreiber mit der Bezeichnung „/dev/crypto“ notwendig, der dann die Hardwarebeschleunigung der von OpenSSL übergebenen Krypto-Operationen durchführt.

Dieser Gerätetreiber zur Behandlung asynchroner Verschlüsselungsoperationen existiert jedoch noch nicht im aktuellen Linux Kernel. Die Lösung dieses Defizits verspricht das OpenBSD/FreeBSD Cryptographic Framework (OCF)[6], welches als Portierung für Linux erhältlich ist.

Mit dem OCF werden zwei universelle Schnittstellen für die hardware-beschleunigte Ausführung krytographischer Operationen bereitgestellt:

1. die „producer“-API zur Anbindung von Treibern für Krypto-Geräte
2. die „consumer“-API zur Ausführung kryptographischer Operationen

Die Schnittstelle „producer“ dient der Registrierung eines Gerätetreibers zur Beschleunigung kryptographischer Operationen. Neben Treibern für verschiedene externe Verschlüsselungs-Erweiterungskarten existiert auch ein Gerätetreiber für die Verwendung des im IXP425 eingesetzten Krypto-Beschleunigers. Nachdem ein Gerätetreiber registriert wurde, werden sämtliche kryptographischen Operationen vom OCF an die Hardwarebeschleunigung weitergeleitet.

Diese Krypto-Operationen können über die Schnittstelle „consumer“ aufgerufen werden. Sie stellt Befehle für das Erstellen einer neuen Verschlüsselungs-Sitzung, crypto_newsession(), sowie zum Ver- und Entschlüsseln crypto_dispatch(), zur Verfügung. Beim Aufruf von crypto_dispatch() werden sämtliche für die Krypto-Operation notwendigen Parameter

[6] Siehe Homepage OCF-Projekt [McC05]

übergeben und die Operation einer Warteschlange hinzugefügt. Danach wird die Ausführung des aufrufenden Programms fortgesetzt und erst nach Beendigung der Krypto-Operation durch eine Rückruffunktion unterbrochen. Durch diese asynchrone Vorgehensweise kann bei geeigneter Verschlüsselungshardware eine deutliche Leistungssteigerung erreicht werden.

Die folgende Abbildung zeigt den beispielhaften Ablauf einer Krypto-Operation:

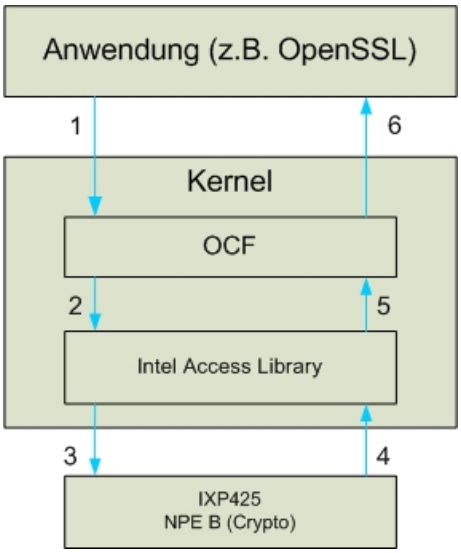

Abbildung 5.6: **Asynchrone Verschlüsselung mit OCF**
1. Absetzen des Aufrufes an das OCF
2. Weiterleitung an die Intel Access Library
3. Ausführung der Operation durch die NPE B des IXP425
4. Rückgabe der verschlüsselten Daten
5. Weiterleitung des Ergebnisses an das OCF
6. Aufruf der „Callback“-Funktion der Anwendung

Bei Vorhandensein mehrerer Beschleunigungs-Prozessoren kann zusätzlich eine Lastverteilung vorgenommen werden - das OCF steuert dabei automatisch die Vergabe der Krypto-Operationen an die zur Verfügung stehenden Prozessoren.

Ein Programm, das von der Hardwarebeschleunigung Gebrauch machen möchte, muss nun statt der CryptoAPI des Kernels die Schnittstelle von OCF nutzen. Aufgrund der Universalität der Schnittstellen des OCF braucht bei einer Veränderung der Hardware, beispielsweise bei Verwendung einer neuen Beschleunigungskomponente, nur ein zusätzlicher Hardware-Treiber für das OCF eingebunden werden - das Anwendungsprogramm muss nicht angepasst werden.. Leider existieren mit OpenSSL und OpenSSH jedoch nur zwei Applikationen, die von dem OpenBSD Cryptographic Framework Gebrauch machen.

Zum Kompilieren muss das OCF in die Quelltextstruktur des Kernels eingearbeitet werden. Dafür wurde im OpenEmbedded ein neuer Kernel mit dem Namen „openixp-kernel-crypto“ erstellt. Danach kann das Krypto-Framework als Kernelmodul kompiliert werden. Dabei werden folgende Module erzeugt:

- ocf.ko
- cryptodev.ko
- cryptosoft.ko
- ixp4xx.ko

Das Kernelmodul „ocf“ ist verantwortlich für die Weiterleitung asynchroner Krypto-Operationen an die Hardwarebeschleunigung, das Modul „cryptodev“ stellt diese anderen Programmen als Gerät „/dev/crypto“ zur Verfügung. Über den Gerätetreiber „ixp4xx“ wird nun die API der Intel Access Library verwendet, um die zu verschlüsselnden Daten an die NPE des IXP425 weiterzuleiten. Im Falle der Nichtverfügbarkeit der Hardwarebeschleunigung nutzt das Framework automatisch das Modul „cryptosoft“, um die standardmäßigen Kernel-Funktionen zur Verschlüsselung zu benutzen.

Trotz des komplizierten Aufrufmechanismus zwischen Kernel, Intel Access Library und OpenBSD Cryptographic Framework kann mit Hilfe dieser Technologie ein deutlicher Zuwachs an Verschlüsselungsleistung vor allem bei großen Paketen erreicht werden.

Eine einfachere Möglichkeit zur Nutzung der Hardwarebeschleunigung durch OpenSSL existiert im Moment leider nicht.

5.3.6 IPSec für Virtuelle Private Netzwerke

Wie im Abschnitt 5.3.2 bereits erwähnt, ist die Verschlüsslung von IP-Paketen durch das IPSec-Protokoll Bestandteil des aktuellen Kernels. Dieses Kernelmodul trägt die Bezeichnung NETKEY und wird durch die Zeile CONFIG_NET_KEY=y in der Kernelkonfiguration aktiviert.

Zur Konfiguration und zum Aufbau von IPSec-Verbindungen ist jedoch eine zusätzliche Software notwendig.

Aus folgenden Gründen fiel hierbei die Wahl auf das Openswan-Projekt:

1. Unterstützung des 2.6er Kernels mit NETKEY
2. Alternativ dazu eigene IPSec-Implementierung (KLIPS)
3. Authentifikation mit X.509-Zertifikaten
4. Kompatibilität mit nahezu allen anderen IPSec-Implementierungen
5. Großer Verbreitungsgrad
6. Umfangreiche Dokumentation und Support-Möglichkeiten

OpenEmbedded liefert im Repository eine Paketbeschreibung für Openswan in der Version 2.2, welche sich jedoch als fehlerhaft im Betrieb auf dem IXDPG425 herausstellte. Durch das Anlegen eines neuen Paketes für Openswan in der aktuellsten Version (2.3) konnte dieser Fehler behoben und damit die Funktionsfähigkeit von IPSec hergestellt werden.

Ein weiteres Problem stellt die Anbindung von Microsoft Windows Betriebssystemen an den IPSec-Server dar. Zwar existiert in den aktuellen Versionen von Windows 2000 und XP eine IPSec-Implementation, jedoch kann diese nur mit großem Aufwand manuell konfiguriert werden. Somit ist die Installation eines VPN-Clients zum Verwenden von nativen IPSec notwendig. Eine wesentlich einfachere Möglichkeit bietet jedoch das Layer2 Transport Protokoll (L2TP), das direkt in Windows integriert ist. Somit kann ein VPN-Tunnel

als neue DFÜ-Verbindung konfiguriert werden. Auf Seite des Servers wird dafür aber eine Implementation des L2TP-Protokolls benötigt - dies leistet beispielsweise die Software RP-L2TP[7]. Da im Repository von OpenEmbedded bisher noch kein L2TP-Dämon vorhanden war, wurde dafür ein neues Paket mit dem Namen „rp-l2tp“ erstellt. Zusätzlich benötigt diese Art des Verbindungsaufbaus einen PPP-Dämon, der sich durch OpenEmbedded einfach portieren lässt.

Mit der Kombination aus Openswan, L2TP und PPP können von Windows-PCs ohne zusätzliche Softwarekomponenten IPSec-Verbindungen initiiert werden. Dabei sorgen sowohl X.509-Zertifikate als auch eine Kombination aus Benutzername und Passwort für eine doppelte Authentifikation. Die Konfiguration der Seite des Clients soll im späteren Verlauf dieser Arbeit präsentiert werden.

5.3.7 Unterstützung von IEEE 802.11i (WPA, WPA2)

Wie bereits erwähnt werden in dem Madwifi-Treiber nur die rudimentären Sicherheitsfunktionen der WEP-Sicherheitsarchitektur implementiert. Um trotzdem einen sicheren Authentifizierungs- und Verschlüsselungsmechanismus nutzen zu können, wird ein zusätzlicher „Authenticator“ benötigt. Diese Funktion kann über die Open Source Software „Hostap-Daemon“ (Hostapd) übernommen werden.

Hostapd ist ein im Benutzerbereich agierender Hintergrunddienst für Linux, der basierend auf verschiedenen WLAN-Treibern (unter anderem Madwifi) zusätzliche Authentifikationsdienste bietet.

Folgende Funktionen lassen sich realisieren:

- WPA-PSK („WPA-Personal“)
- WPA mit EAP („WPA-Enterprise“)
- Schlüsselmanagement für CCMP, TKIP, WEP40, WEP104
- IEEE 802.11i, RSN, WPA2
- RADIUS Accounting
- RADIUS Authentification über EAP

Das Extensible Authentication Protocol (EAP) kann mit einer Vielzahl von unterschiedlichen Authentifikations-Methodenbetrieben werden. Die folgenden Methoden zur Authentifikation werden derzeit von Hostapd unterstützt:

- EAP-TLS
- EAP-PEAP/MSCHAPv2 (PEAPv0 und PEAPv1)
- EAP-PEAP/GTC (PEAPv0 und PEAPv1)

[7] Siehe Homepage RP-L2TP Projekt [Pro04b]

- EAP-PEAP/MD5-Challenge (PEAPv0 und PEAPv1)
- EAP-TTLS/EAP-MD5-Challenge
- EAP-TTLS/EAP-GTC
- EAP-TTLS/EAP-MSCHAPv2
- EAP-TTLS/MSCHAPv2
- EAP-TTLS/MSCHAP
- EAP-TTLS/PAP
- EAP-TTLS/CHAP
- EAP-SIM

Das Hostapd-Paket bietet damit vielfältige Möglichkeiten für die Authentifikation von Benutzern, die in den unterschiedlichen Einsatzbereichen unterschiedliche Vor- und Nachteile aufweisen. Die für diese Arbeit relevanten Authentifikations-Methoden wurden bereits innerhalb des Kapitels 2.8 diskutiert.

Besonders vorteilhaft erweist sich bei kleinen Installationen der integrierte „Authentication Server“, der im Modus WPA-Enterprise einen externen Radius-Server ersetzt. Dabei übernimmt der Hostap-Dämon sowohl die Rolle des „Authenticator“ als auch die des „Authentication Servers“.

Verfügt das Unternehmen jedoch bereits über einen Radius-Server, kann dieser selbstverständlich auch als Authentication-Server eingesetzt werden.

Der Hostap-Dämon existiert sowohl in einer stabilen Produktivversion (Versionsnummer 0.3.9), als auch in einer „unstable“-Entwicklerversion (0.4.3).

Da in der Entwicklerversion mehrere Funktionen enthalten sind, die in dieser Arbeit von Interesse sind und außerdem eine bessere Unterstützung des gewählten WLAN-Treibers versprochen wird, wurden sowohl die stabile als auch die unstabile Version auf dem IXDPG425 getestet.

Das Paket „Hostap-Daemon“ wird von OpenEmbedded bereitgestellt, jedoch muss hierbei eine Konfigurationsdatei angepasst werden, um dem Compiler den Pfad des verwendeten Madwifi-Treiber mitzuteilen.

5.3.8 Firewall

Die Funktionalität einer Firewall ist im aktuellen Linux Kernel vollständig integriert und muss nur noch über die Kernelkonfiguration aktiviert werden. Dafür existiert neben der Option „CONFIG_NETFILTER“ eine lange Liste weiterer Parameter, um verschiedene Filteroptionen aktivieren bzw. deaktivieren zu können.

Zum Konfigurieren der Filterregeln wird ein Paket namens „iptables“ benötigt, das mit OpenEmbedded kompiliert werden kann.

Jedoch kann sich in einem komplexen System die manuelle Einrichtung der Firewall-Regeln vor allem für ungeübte Administratoren zu einer schwierigen Aufgabe entwickeln. Deshalb wurde nach einer Lösung zur komfortablen Verwaltung der Firewall gesucht. Das dabei gefundene Open Source Paket „shorewall“ besteht aus verschiedenen Textdateien, in denen einfach und leicht verständlich sämtliche Einstellungen der Firewall vorgenommen werden können. Aus diesen Textdateien werden beim Start des Shorewall-Skriptes Regeln für das Programm „iptables“ erzeugt und diese dann dem Kernel übergeben.

Auch Shorewall kann über die Entwicklungsumgebung OpenEmbedded einfach an den IXDPG425 angepasst werden, jedoch sollte dabei eine Erweiterung der Paketbeschreibung auf die aktuellste Version 2.4.0 dieser Software vorgenommen werden.

Als problematisch erweist sich beim Einrichten der Firewall jedoch die Einbindung des IPSec-Protokolls in den Kernel. Während bei der Verwendung eines 2.4er Kernels noch ein Netzwerkinterface ipsec0 vorhanden ist, wurde hierauf in dem aktuellen Kernel verzichtet.

Somit passiert jedes eintreffende, mit IPSec verschlüsselte Paket genau zweimal die Firewall: zuerst als verschlüsseltes, danach als unverschlüsseltes Paket. Dadurch lassen sich nun keine Regeln formulieren, die bestimmte Dienste (beispielsweise den Zugriffe auf das Unternehmensnetzwerk) nur für Nutzer von IPSec-Verbindungen ermöglichen.

Eine Abhilfe für diese Problematik schafft ein zusätzlicher Filteralgorithmus, dessen Einbindung in den Kernel in einer online verfügbaren Beschreibung[8] vorgestellt wird. Aufgrund des hohen Umfangs dieser Quellcode-Anpassung wird auf eine ausführliche Darstellung an dieser Stelle verzichtet.

5.3.9 Zusätzliche Netzwerkdienste

Neben den bisher vorgestellten Programmen, die nahezu alle gewünschten Sicherheitsfunktionen des WLAN Access Points realisieren, sind noch weitere Netzwerkdienste notwendig, die vor allem im Routerbetrieb die Arbeit mit dem IXDPG425 vereinfachen.

So ist es beispielsweise wünschenswert, dass drahtlose Computer beim Zugriff auf den WLAN Router automatisch eine IP-Adresse aus dem internen Netz erhalten. Diese Aufgabe erfüllt ein Server, der das Dynamic Host Configuration Protocol (DHCP) implementiert. Häufig wird im Internet der ISC DHCP-Server genutzt, der jedoch aufgrund seiner Komplexität die Anforderungen an Ressourcensparsamkeit nicht erfüllt. Eine Alternative stellt das Open Source Projekt Dnsmasq dar, welches neben einem kleinen und leichtgewichtigen DHCP-Server auch die Funktion der Weiterleitung von DNS-Anfragen realisiert.

Aufgrund des Fehlens einer Echtzeituhr muss außerdem bei jedem Start des Access Points die Uhr gestellt werden, um beispielsweise Gültigkeitsdauer eines X.509-Zertifikates korrekt prüfen zu können. Um das Einstellen der korrekten Uhrzeit zu automatisieren, kann das Network Time Protocol (NTP) genutzt werden. Eine Implementation dieses Protokolls

[8] HowTo: IPSEC using Linux Kernel 2.6 [Eas05]

befindet sich im Paket „ntp“, welches wie „dnsmasq“ automatisch über OpenEmbedded für den IXDPG425 portiert werden kann.

Zur entfernten Administration des Access Points sollte zusätzlich ein Dienst installiert werden, der einen Zugang per Secure Shell (SSH) ermöglicht. Eine leichtgewichtige Implementation des SSH-Protokolls bietet das Paket „dropbear“.

5.3.10 Netzwerktest und -diagnose

Um ein- und ausgehende Pakete auf dem WLAN Access Point auslesen zu können, beispielsweise zum Nachweis der Paketverschlüsselung, wird während der Testphase ein Paketmonitor benötigt. Außerdem soll auf dem Router eine Software zur Messung der Netzwerk-Übertragungsgeschwindigkeit installiert werden, um den Einfluss der verschiedenen Verschlüsselungsoptionen auf den Datendurchsatz zu bestimmen.

Durch eine Recherche im Repository von OpenEmbedded konnte das freie Paket „tcpdump“ zum Mitschneiden von IP-Paketen sowie die Software „netperf“ zur Messung der Übertragungsgeschwindigkeit gefunden werden.

Diese beiden Pakete werden zumindest während der Testphase auf dem IXDPG425 benötigt.

Kapitel 6

Praktische Umsetzung der WLAN Distribution für den IXDPG425

Nachdem im voherigen Kapitel verschiedene Softwaremodule zur Realisierung der gewünschten Einsatzszenarien vorstellt wurden, sollen diese nun für die Zielplattform IXDPG425 kompiliert und konfiguriert werden. Die Ergebnisse des Funktionstests sowie daraus resultierende Änderungen sind ebenfalls Gegenstand dieses Abschnittes.

6.1 Kompilierung der Module

Der Kompilier-Vorgang wird weitestgehend durch die Entwicklungsumgebung OpenEmbedded unterstützt. Zum Zusammenfassen mehrerer Pakete zu einer Distribution können dabei Meta-Pakete im Verzeichnis „./packages/meta“ definiert werden. Für den IXDPG425 wurde ein neues Meta-Paket „openixp-image“ erstellt, welches neben dem im Kapitel 5.3 vorgestellten Paketen noch Module für den allgemeinen Funktionsumfang einer Linux-Distribution enthält. Beim Starten von Bitbake, mit dem Namen des Meta-Moduls als Parameter, werden zuerst alle darin enthaltenen Softwaremodule kompiliert und in Verzeichnis „rootfs“ installiert. Danach erfolgt die automatische Erstellung eines Wurzeldateisystems (Root-Filesystem, rootfs) in dem Format „ext2“, das dann in den Speicher des IXDPG425 übertragen werden kann.

Die in der Tabelle 6.1 dargestellten Softwaremodule konnten durch OpenEmbedded mit den vorhandenen bzw. selbst erstellten Paketbeschreibungen automatisch kompiliert werden.

Die in der Spalte „OE“ gekennzeichneten Module konnten dabei direkt aus dem OpenEmbedded Repository übernommen werden, für allen anderen mussten eigene Paketbeschreibungen angelegt werden.

Während der Kompilier-Vorgang aufgrund der Unterstützung durch OpenEmbedded häufig unproblematisch durchgeführt werden konnte, ergaben sich beim Test der Softwaremodule in einigen Fällen schwerwiegende Probleme. Diese sollen nun beispielhaft dargestellt werden.

Beschreibung	Paketname	Version	OE
Toolchain	binutils gcc glibc	2.14 3.3.4 2.3.2	X X X
Kernel	openixp-kernel	2.6.11.2	
Allgemeine Dienstprogramme	busybox	1.00	X
Intel Access Library ohne Krypto-Unterstützung	ixp4xx-csr	1.4	X
Intel Ethernet Driver	ixp425-eth	1.1	X
OpenSSL	openssl	0.9.7e	X
IPSec: Openswan	openswan	2.3.0	
IPSec: zusätzliche Dienstprogramme	ipsec-tools	0.5	X
Layer 2 Transport Protokoll	rp-l2tp	0.4	
IEEE 802.11i Authenticator	hostap-daemon	0.4.3 (unstable) 0.3.9 (stable)	
Firewall Dienstprogramm	iptables	1.3.1	
Firewall Konfigurationsprogramm	shorewall	2.4.0	
DHCP Server, DNS Weiterleitung	dnsmasq	2.22	X
SSH Server	dropbear	0.45	X
Bridge Dienstprogramm	bridge-utils	1.0.4	X
PPP Server	ppp	2.4.3	X
WLAN Dienstprogramme	wireless-tools	28	X
Paketmonitor	tcpdump	3.8.3	X
Network Time Protokoll	ntp	4.1.2	X
Performance Analyse	netperf	2.3	X

Tabelle 6.1: Module der WLAN Distribution

6.1.1 Kompilieren des Madwifi-Treibers

Die größten Probleme bei der prototypischen Realisierung der WLAN Distribution wurden durch den Treiber für den WLAN-Adapter hervorgerufen. Sowohl Kompatibilitätseinschränkungen aufgrund der Verwendung des binären „Hardware Access Layer“ als auch das frühe Entwicklungsstadium dieses Treibers verursachten große Schwierigkeiten.

So wurde anfänglich die Verwendung eines aktuellen GCC-Compilers der Version 3.4 für die Erstellung der Distribution vorgesehen. Damit ließ sich Madwifi problemlos kompilieren, jedoch konnten keine TCP/IP-Pakete über die drahtlose Verbindung übertragen werden. Erst die vollständige Umstellung auf die Verwendung von GCC 3.3.4 ermöglichte die Lösung dieses Problems.

Weitere Schwierigkeiten ergaben sich durch die Existenz zweier unterschiedlicher Entwicklungslinien dieses Treibers. Neben der Hauptentwicklungslinie existiert ein davon abgespaltenes Projekt namens „madwifi-bsd“, welches aktuelle Änderungen aus dem FreeBSD Atheros-Treiber in den Linux-Treiber integriert.

Während die Standardfunktionalität durch beide Treiberversionen gewährleistet wurde, konnten beim Integrationstest schwerwiegende Kompatibilitätsprobleme des originalen

Madwifi-Treibers nachgewiesen werden. Diese traten vor allem bei der Anwendung von IEEE 802.11i mit dem Hostap-Dämon auf. Eine Umstellung auf die BSD-Version ermöglichte die Lösung dieser Probleme.

6.1.2 Integration der Krypto-Unterstützung in die Intel Access Library

Im Kapitel 5.3.3 wurden bereits die zwei verschiedenen Versionen der Intel Access Library vorgestellt. Um auch die kryptographischen Komponenten in das Kernel-Modul zu integrieren, musste der dafür vorhandene Quellcode manuell in die durch OpenEmbedded bereitgestellte Access Library eingefügt und das Makefile angepasst werden. Zudem wurde der von Intel bereitgestellte Microcode für die kryptographischen Operationen der NBE B in den Ethernet-Treiber übernommen.

6.1.3 Integration des OCF in den Linux-Kernel

Um das OpenBSD Cryptographic Framework als Kernel-Modul kompilieren zu können, muss dessen Quellcode direkt in den Kernelquellcode eingearbeitet werden. Ein beiliegender Patch sorgt für die automatische Anpassung der notwendigen Dateien. Dabei wurde ein neues Kernel-Paket „openixp-kernel-crypto“ angelegt, um im Falle des Nichtfunktionierens des OCF auf den bisher funktionsfähigen Kernel „openixp-kernel“ zurückgreifen zu können.

Nach dem Neukompilieren der Kernel-Module wurden die neu erstellten Dateien ocf.ko, cryptodev.ko, cryptosoft.ko und ixp4xx.ko in das Dateisystem des IXDPG425 übernommen und beim Bootvorgang geladen.

6.1.4 Kompilierung und Test von Dnsmasq

Analog zu den bereits beschriebenen Softwaremodulen konnte auch Dnsmasq automatisch durch OpenEmbedded für den IXDPG425 kompiliert werden. In umfangreichen Tests wurde jedoch festgestellt, dass das Serverprogramm zum Beantworten von DHCP- und DNS-Anfragen keine Reaktion auf eingehende Pakete liefert. Durch die direkte Kommunikation mit den Entwicklern wurde der Fehler auf den Kompiliervorgang eingegrenzt und konnte durch Entfernen des Optimierungsparameters „-O2“ gelöst werden. Dieses und ähnliche in diversen Mailinglisten diskutierten Bespiele, zeigen, dass der verwendete GCC-Compiler nicht immer einen auf dem IXDPG425 lauffähigen Binärcode erzeugt. Häufig sind für das Portieren kleine Änderungen im Makefile oder im Quellcode notwendig.

6.1.5 Erweiterung zur Unterstützung mehrerer Benutzergruppen

Eine durch die bisher vorgestellten Softwaremodule nicht ermöglichte Funktionalität ist die in der Aufgabenstellung geforderte Unterstützung mehrerer Benutzergruppen. Dies bedeutet, dass im WLAN Access Point jedem Benutzer eine Benutzergruppe mit jeweils unterschiedlichen Zugriffsrechten zugeordnet werden kann. Im Kapitel 5.2.5 wurde bereits die Multi-BSS-Technologie, die in letzter Zeit von einigen Herstellern implementiert wird, vorgestellt. Da der MADWIFI-Treiber im Moment jedoch noch keine Unterstützung dieser Technologie bietet, musste eine alternative Möglichkeit zu einer demonstrativen Umsetzung dieser Funktionalität gesucht werden.

Dabei entstand im Verlauf dieser Arbeit die Idee, basierend auf der Identifikation des Benutzers eine dynamische Anpassung der Firewall-Regeln durchzuführen. Somit kann beispielsweise zum Schutz des lokalen Netzes vor Gästen ein spezieller IP-Adressbereich für diejenige MAC-Adresse gesperrt werden, mit der sich ein Gast anmeldet.

Zur Implementation dieser Funktion ist die Anpassung des Authenticators - in der Distribution die Komponente „Hostap-Dämon“ - notwendig, da nur sie über die Identität des angemeldeten Benutzers verfügt. Der Authenticator muss hierbei dahingehend verändert werden, dass bei jedem Anmeldevorgang ein Shell-Skript aufgerufen wird, welches dann eine auf dem Benutzernamen basierende Anpassung der Firewall-Regeln durchführt. Die hierfür entwickelte Lösung hat ausschliesslich demonstrativen Charakter und bietet keineswegs ausreichenden Komfort zum Einsatz in Produktivsystemen.

Zur Manipulation der Firewall-Regeln mit dem Programm „iptables“ beim Verbindungsaufbau und zum Eintrag der Aktion in die Log-Datei wurde das folgende Skript erstellt:

Listing 6.1: Skript setguest

```
1  #!/bin/sh
2
3  case "\$1" in
4   gast)
5    logger "Benutzername" "$1" "mit MAC " "$2" "wird geblockt."
6    iptables -I INPUT -j REJECT -m mac --mac-source "$2"
7                  -d 192.168.0.0/255.255.255.0
8    ;;
9   *)
10   ;;
11 esac
```

Das Shellskript wird dabei mit zwei Parametern aufgerufen:

1. Benutzername des assoziierten WLAN-Clients

2. MAC-Addresse des Clients

Über eine auf dem Benutzernamen basierende Auswahlbedingung wird beim Start des Skriptes der jeweilige Befehl zur Manipulation der Firewall aufgerufen und damit beispielsweise ein spezieller IP-Adressbereich für die im zweiten Parameter übermittelte IP-Adresse gesperrt. Für jeden Benutzernamen, dessen Firewall-Regeln von der Standard-Konfiguration abweichen sollen, muss hierbei eine gesonderte Auswahlbedingung definiert werden.

Beim Verlassen des Funknetzes müssen die in der Anmeldephase vorgenommenen Änderungen jeweils rückgängig gemacht werden. Dafür wird das folgende Skript verwendet:

Listing 6.2: Skript delguest

```
#!/bin/sh

case "\$1" in
 gast)
  logger "Sperre für Benutzername " "$1" " mit
                MAC " "$2" "wird aufgehoben."
  iptables -D INPUT -j REJECT -m mac --mac-source "$2"
                -d 192.168.0.0/255.255.255.0
  ;;
 *)
  ;;
esac
```

Da der Aufruf der Skripte stets beim Aufbau sowie beim Abbau einer WPA-Verbindung durchgeführt werden muss, wurde der Quelltext des Hostap-Dämons um diese Aufrufe erweitert. Die Verbindungssteuerung bei Verwendung von WPA ist als Zustandsautomat in der Quelltextdatei „wpa.c“ implementiert. Befindet sich der Automat im Zustand „PTKINITDONE“ ist die Verbindung erfolgreich aufgebaut. Somit wurde an dieser Stelle der Aufruf des Skriptes „setguest“ folgendermaßen eingefügt:

Listing 6.3: wpa.c Zeilen 2380-2383

```
char parameter[100];
snprintf (parameter,100, "/etc/hostapd/setguest \%s " MACSTR,
        sm->sta->eapol_sm->identity, MAC2STR(sm->sta->addr));
FILE *file = popen(parameter, "r");
pclose (file);
```

Hierbei wird zuerst die aufzurufende Befehlszeile mitsamt Parametern zusammengestellt, „/etc/hostapd/setguest“ bezeichnet das zu startende Skript, „sm->sta->eapol_sm->identity“ die Identität des Benutzers und „MAC2STR(sm->sta->addr)“ die MAC-Adresse des Clients. Über den Befehl „popen“ wird dieser zusammengesetzte Befehl gestartet.

Analog gestaltet sich der Aufruf des Skriptes beim Beenden der WLAN-Verbindung im Zustand „DISCONNECTED“. Statt dem Skript „setguest“ wird nun lediglich das Pendant „delguest“ mit den beiden Parametern gestartet.

Die mit den beschriebenen Veränderungen kompilierte Version des Hostap-Dämon wurde unter der Bezeichnung „hostapd-gast“ in das Dateisystem übernommen und erfolgreich getestet. Wie bereits geschildert, soll diese Programmanpassung nur demonstrativ die Vorgehensweise zur Erweiterung des Authenticators um die gewünschte Funktionalität darstellen. Jedoch wird hierbei eindrucksvoll ersichtlich, mit welch einfachen Mitteln Anpassungen der genutzten Open Source Software möglich sind.

6.2 Konfiguration und Test des WLAN Access Points

Nachdem im vorangegangen Abschnitt sämtliche für einen WLAN Access Point und -Router notwendigen Module kompiliert wurden, liegt nun neben einem funktionsfähigen Kernel auch ein Wurzel-Dateisystem „rootfs“ vor. Durch OpenEmbedded wird das „rootfs“ automatisch in einem komprimierten Format gespeichert und kann nun durch das Trivial File Transfer Protocol (TFTP)-Protokoll in den IXDPG425 übertragen werden. Um Änderungen am Dateisystem auch nach einem Neustart des IXDPG425 erhalten zu können, ist es notwendig, das „rootfs“ in den nichtflüchtigen Speicher (Flash-Speicher) des IXDPG425 zu kopieren. Die Größe des Flash-Speichers ist mit 16 MByte jedoch zu klein für die große Anzahl der verwendeten Softwaremodule und Bibliotheken. Daher wurde die Entscheidung getroffen, das Wurzel-Dateisystem auf einem USB-Stick mit der Größe von 128 MByte zu speichern und nach dem Laden des Kernels von dort aus einzubinden. Auf diese Weise ist in der Testphase auch das Speichern umfangreicher Log- und Debug-Informationen möglich.

Zum Zugriff auf das Dateisystem des IXDPG425 wurde ein NFS-Server (Network File System) installiert. Somit konnten während der Laufzeit Änderungen an den Konfigurationsdateien vorgenommen werden.

Die Konfiguration der verwendeten Softwaremodule wird zum Großteil durch Textdateien im Verzeichnis „/etc“ durchgeführt. Die Tabelle 6.2 zeigt die hierfür erstellten Konfigurationsdateien, die kommentiert auch im Anhang zu finden sind.

Modul	**Konfigurationsdatei**
Dnsmasq	/etc/dnsmasq.conf
IPSec (Openswan)	/etc/ipsec/ipsec.conf
L2TP-Dämon	/etc/l2tp/l2tp.conf
PPP-Dämon	/etc/ppp/options /etc/ppp/options.lns
Hostap-Dämon	/etc/hostapd/hostapd-intradius.conf /etc/hostapd/hostapd-radius.conf /etc/hostapd/hostapd-psk.conf /etc/hostapd/hostapd.eap-user
Firewall (Shorewall)	/etc/shorewall/*

Tabelle 6.2: Übersicht Konfigurationsdateien

Neben der Konfiguration der einzelnen Module mussten außerdem Startskripte zum Starten und Beenden dieser Programme erstellt und in das Verzeichnis „/etc/init.d/“ kopiert werden. Die Zuordnung der Module zu den verschiedenen Systemzuständen des Betriebssystems (Runlevel) erfolgt über die Verzeichnisse „/etc/rc.S“, „/etc/rc.1“ bis „/etc/rc.6“.

Für den Betrieb der in der WLAN Distribution enthaltenen Sicherheitsfunktionen ist die Existenz einer Zertifizierungsstelle und von ihr ausgestellter Zertifikate notwendig. Der Vorgang der Zertifikatserstellung mit OpenSSL soll im folgenden Kapitel beispielhaft dargestellt werden.

6.2.1 Erstellen von X.509-Zertifikaten

Für die Verwendung von EAP bei der WLAN-Authentifizierung sowie für die Authentifizierung mit IPSec müssen eine Vielzahl von Zertifikaten und Schlüsseln erzeugt und verwaltet werden. Ebenso ist eine Zertifizierungsstelle (Certificate Authority: CA) notwendig, weshalb der Aufbau einer Public Key Infrastruktur (PKI) erfolgen muss. Mit OpenSSL steht eine freie SSL-Implementierung zur Verfügung, die diese Aufgabe leisten kann. Im Falle einer bestehenden PKI kann diese selbstverständlich auch für die Verwaltung der notwendigen Zertifikate im Produktivbetrieb eingesetzt werden.

Erstellen eines CA Stammzertifikates

Um ein neue Zertifizierungsstelle zu erzeugen, muss zuerst ein „Request“ gestellt werden:

```
openssl req -new -x509 -keyout newreq.pem -out newreq.pem -days 730
```

Dabei muss ein Passphrase eingeben werden, der den privaten Schlüssel der Zertifizierungsstelle schützt. Außerdem erfolgt die Abfrage einiger beschreibender Informationen.

Der folgende Befehl nutzt ein bei Openssl mitgeliefertes Skript, welches aus dem „Request“ eine neue CA erstellt:

```
echo "newreq.pem" | /usr/lib/ssl/misc/CA.pl -newca
```

Nach dem Starten des Skriptes existiert ein neues Verzeichnis demoCA, in welchen sich alle für die Verwaltung der CA notwendigen Dateien befinden.

Um das Stammzertifikat auch in anderen Betriebssystemsumgebungen nutzen zu können, konvertieren die folgenden 3 Befehle es in verschiedene Formate:

```
openssl pkcs12 -export -in demoCA/cacert.pem -inkey newreq.pem -out root.p12 \
-cacerts
```

```
openssl pkcs12 -in root.p12 -out root.pem
```

```
openssl x509 -inform PEM -outform DER -in root.pem -out root.der
```

Nach Beendigung dieser Schritte steht das Stammzertifikat in verschiedenen Formaten zur Verfügung: Die Datei demoCA/cacert.pem wird durch den „Authenticaton Server“ für das Überprüfen von Benutzerzertifikaten verwendet; die Datei root.der muss auf den PCs der Benutzer installiert werden, um eine gegenseitige Authentifizierung gewährleisten zu können; die drei Dateien root.p12, root.pem und demoCA/private/cakey.pem enthalten den privaten Schlüssel der Zertifizierungsstelle - verwendet für das Unterschreiben neuer Zertifikate - und müssen daher geheim gehalten werden.

Erstellen des Server-Zertifikates

Wie zuvor muss zum Erzeugen eines Zertifikates für den Server zuerst ein neuer „Request“ durchgeführt werden. Hierbei werden verschiedene beschreibende Informationen eingegeben; wichtig ist vor allem die Eingabe eines „Common Name“, in diesem Fall beispielsweise „Server“.

```
openssl req -new -x509 -keyout newreq.pem -out newreq.pem -days 730
```

Im folgenden Schritt wird durch die gerade erstellte CA der Request unterschrieben und damit ein Serverzertifikat erstellt. Vorraussetzung dafür ist das Vorhandensein einer Datei namens xpextensions, welche die folgenden Zeilen beinhaltet:

```
[ xpclient_ext]
extendedKeyUsage = 1.3.6.1.5.5.7.3.2
[ xpserver_ext ]
extendedKeyUsage = 1.3.6.1.5.5.7.3.1
```

Mit diesen Einstellungen wird die gegenseitige Authentifikation zwischen Server und Client für Windows XP-Clients ermöglicht. Nun kann mit dem folgenden Befehl die Erstellung des Zertifikats durchgeführt werden. Dabei wird das Passwort für den privaten Schlüssel der Zertifizierungsstelle benötigt.

```
openssl ca -policy policy_anything -out newcert.pem -extensions xpserver_ext \
-extfile xpextensions -infiles newreq.pem
```

Mit weiteren Befehlen wird das Zertifikat in verschiedene Formate exportiert.

```
openssl pkcs12 -export -in newcert.pem -inkey newreq.pem  -out cert-srv.p12 \
-clcerts

openssl pkcs12 -in cert-srv.p12 -out cert-srv.pem
```

Die Datei cert-srv.pem, welche auch den privaten Schlüssel enthält, wird nun als Serverzertifikat für Openswan bzw. den im Hostap-Dämon integrierten „Authentication Server“ festgelegt.

Eine weitere benötigte Datei enthält Informationen über die Diffie-Hellmann Schlüsselparameter. Diese Datei lässt sich mit dem folgenden Befehl erstellen:

```
openssl dhparam -check -text -5 512 -out dh
```

Erstellen eines Client-Zertikates

Ähnlich der Erstellung des Server-Zertifikates wird nun für jeden Client (Supplicant) ein separates Zertifikat produziert. Dafür werden die folgenden Befehle ausgeführt:

Erstellen eines neuen Request:

```
openssl req -new -x509 -keyout newreq.pem -out newreq.pem -days 730
```

Unterschreiben des Requests mit dem Schlüssel der CA:

```
openssl ca -policy policy_anything -out newcert.pem -extensions xpclient_ext \
-extfile xpextensions -infiles newreq.pem
```

Exportieren des Zertifikates:

```
openssl pkcs12 -export -in newcert.pem -inkey newreq.pem -out cert-clt.p12
-clcerts
```

```
openssl pkcs12 -in cert-clt.p12 -out cert-clt.pem
```

Die Datei cert-clt.p12 enthält nun das Client-Zertifikat sowie den privaten Schlüssel des Clients in binärer Form und muss nun auf einem sicheren Weg auf den PC des Benutzers kopiert werden.

Bei der Verwendung von Windows als Client-Betriebssystem muss die Datei cert-clt.p12 auf den Client-PC sicher übertragen werden. Wenn Linux als Betriebssystem für den Supplicant genutzt wird, benötigt der Client die im ASCII-Format vorliegende Datei client-clt.pem.

Neben der manuellen Erstellung der Zertifikate per Befehlzeile existieren grafische Benutzeroberflächen für OpenSSL sowie komplexe Produkte zum Aufbau und zur Administration einer Zertifizierungsstelle (CA). Beispiel für eine grafische Benutzeroberfläche ist TinyCA, eine unter X-Window lauffähige Software zum Erstellen, Exportieren und Zurücknehmen von Zertifikaten.

Ein wesentlich komplexeres Programmpaket zum Betreiben einer Zertifizierungsstelle stellt das OpenCA-Projekt[1] zur Verfügung. Dabei wird die Ausstellung von Zertifikaten beginnend beim Registrierungsprozess, über die Überprüfung der Identität, bis hin zur Übergabe des Zertifikates auf organisatorischer und technischer Seite unterstützt.

[1] Vgl. The OpenCA Project [Pro04a]

6.2.2 Konfiguration des WLAN Adapters

Um den WLAN Adapter bezüglich seiner Funkeinstellungen parametrisieren zu können, werden die Dienstprogramme aus dem „wireless-tools“-Paket genutzt.

Die „wireless-tools“ stellen folgende Befehle für die WLAN-Funktionalität unter Linux zur Verfügung:

iwconfig Ähnlich dem bekannten Programm ifconfig, welches zur Konfiguration von Netzwerkgeräten eingesetzt wird, können mit iwconfig spezielle Parameter für drahtlose Netzwerkadapter manipuliert werden. Außerdem stellt iwconfig Informationen über den Zustand des WLAN-Adapters dar.

iwlist Dieses Hilfsprogramm dient zum Anzeigen detaillierter Informationen über ein drahtloses Netzwerkgerät.

iwpriv Hiermit können optionale, häufig treiberabhängige Parameter des WLAN-Adapters verändert werden.

iwspy Iwspy dient zur Darstellung statistischer Informationen im drahtlosen Netzwerk.

Eine Datei namens „/etc/network/interfaces“ sorgt für die Konfiguration sämtlicher Netzwerkadapter und enthält auch die WLAN-spezifischen Parameter.

Beim Startvorgang wird der WLAN-Adapter sofort in den Master-Mode versetzt und arbeitet somit als Access Point. Um die Verbindung zu schützen, kann hierbei bereits ein WEP-Schlüssel festgelegt werden. Für weitergehende Schutzmaßnahmen wird im Anschluss das Programm Hostap-Dämon gestartet, welches die Authentifizierung nach der Norm IEEE 802.11i übernimmt.

6.2.3 Konfiguration des Hostap-Dämon

Für die angestrebten Sicherheitsanforderungen sind die im WLAN-Treiber enthaltenen Protokolle nicht ausreichend. Daher wird beim Hochfahren des Access Points der Hostap-Dämon gestartet, welcher die verschiedenen, im Standard IEEE 802.11i definierten Sicherheitsprotokolle implementiert.

Die Konfiguration dieses Hintergrunddienstes erfolgt dabei über eine Konfigurationsdatei im Verzeichnis „/etc/hostapd/“, in der eine Vielzahl von Parametern verändert werden kann. Neben der Version des Authentifizierungsprotokolls (WPA, WPA2) und der Art der Verschlüsselung (TKIP, CCMP) können sowohl externe Radius-Server zur Authentifizierung als auch ein integrierter Authentication Server verwendet werden. Bei der Anwendung des integrierten Authentifizierungs-Dienstes erfolgt die Vergabe von Benutzernamen und Passwörtern über die Datei hostapd.eap_users. Die für die Authentifizierung notwendigen Zertifikate (Zertifizierungsstelle, Server) müssen dann dem Hostap-Dämon direkt zur Verfügung gestellt werden.

Nutzt man stattdessen einen externen Radius-Server, so wird dem Hostap-Dämon nur dessen IP-Adresse sowie ein gemeinsames Geheimnis zur Kommunikation mitgeteilt. Die Benutzer- und Zertifikatsverwaltung erfolgt dann ausschließlich im Radius-Server.

Zwei Beispiele für die Konfiguration des Hostap-Dämons - sowohl mit integrierten als auch mit externem Authentication Server - befinden sich im Anhang dieser Arbeit.

6.2.4 Test der WLAN Verbindung

Bereits nach dem Initialisieren des Madwifi-Treibers und dem Einstellen des Master-Modus ist der IXDPG425 als WLAN Access Point funktionsfähig und kann durch einen Client erkannt werden. Mit eingeschalteter WEP-Verschlüsselung können sowohl Linux- als auch Windows-Clients ohne zusätzliche Software eine Verbindung zu diesem Access Point aufbauen.

Nach dem Start des Hostap-Dämons ist der Verbindungsaufbau jedoch nur noch nach erfolgreicher Authentifizierung möglich, weshalb zumindest das Zertifikat der Stammzertifizierungsstelle auf dem Client benötigt wird. Außerdem wird auf dem Client eine Software benötigt, welche die Funktionalität des „Supplicant“ im EAP-Protokoll implementiert. Während im Betriebssystem Windows XP Service Pack 2 dies bereits integriert ist, sind für ältere Windows-Versionen aber auch Linux zusätzliche Softwaremodule notwendig.

Unter Linux kann hierfür das Paket „wpa_supplicant“[2] genutzt werden, welches basierend auf dem angewendeten WLAN-Treiber eine Authentifizierung nach den Protokollen der Norm IEEE 802.11i durchführt.

Bei Verwendung von Windows-Betriebssystemen ist die Kompatibilität mit den aktuellsten Sicherheitsprotokollen stark abhängig vom Treiber des Hardware-Herstellers. Zu Beginn der Arbeit existierte für die auf dem Atheros-Chipsatz basierenden WLAN-Adapter keine Treibersoftware, mit welcher ein Verbindungsaufbau bei Verwendung von WPA bzw. WPA2 möglich war. Erst kurz vor Beendigung der praktischen Realisierung konnte mit dem Atheros-Treiber der Version 4.1.2.25 eine Software gefunden werden, die sowohl mit Windows 2000 als auch mit Windows XP sämtliche in dieser Arbeit genutzten Sicherheitsprotokolle und Verschlüsselungsalgorithmen ermöglicht.

Dabei konnten unter Verwendung des internen Authentication Servers folgende Funktionalitäten erprobt werden:

- EAP-Methoden: EAP-TLS, EAP-TTLS/MSCHAPV2, PEAP/MSCHAP
- Verschlüsselungsalgorithmen: TKIP, CCMP
- Sicherheitsprotokolle: WPA, WPA2/RSN
- Betriebssystemkompatibilität: Linux, Windows 2000, Windows XP

[2] Vgl. Linux WPA/WPA2/IEEE 802.1X Supplicant [Mal05b]

Hierbei lässt sich jedoch feststellen, dass einige dieser Funktionen nur mit einem geeigneten WLAN-Adapter im Client genutzt werden können. So kann aufgrund der Komplexität des im CCMP-Protokoll eingesetzten Verschlüsselungsstandards AES dieser nur mit aktuellen WLAN-Endgeräten (z.B. aktuelle Atheros-Chipsätze) verwendet werden. Nutzt man stattdessen veraltete WLAN-Adapter, ist häufig nur die Anwendung des auf dem WEP-Protokoll basierenden Verschlüsselungsalgorithmus TKIP möglich.

Auch die Funktionsfähigkeit mit einem externen Radius-Server konnte erfolgreich nachgewiesen werden. Der auf Open Source-Basis erhältliche Freeradius-Server[3] überzeugte hierbei durch schnelle Installation und relativ einfache Konfiguration.

6.2.5 Konfiguration und Test von IPSec

Sämtliche für den Betrieb von IPSec benötigten Konfigurationsdateien befinden sich unterhalb des Verzeichnisses „/etc/ipsec“. Neben einer Datei „ipsec.secrets“, in welcher die Konfiguration der privaten Schlüssel erfolgt, wird eine Konfigurationsdatei „ipsec.conf“ benötigt. In dieser werden sowohl die globalen als auch die verbindungsspezifischen Parameter für IPSec definiert. Die auf dem IXDPG425 getestete Konfiguration, welche auch im Anhang einsehbar ist, legt eine Vielzahl von Verbindungen für unterschiedliche Einsatzbereiche fest.

wlan Native[4] IPSec-Verbindungen über den WLAN-Adapter.

wlan-l2tp-2k Von einem Windows 2000-Client ausgehende IPSec-Verbindungen, welche das Layer2-Transport-Protocol nutzen.

wlan-l2tp-xp Von einem Windows XP-Client ausgehende IPSec-Verbindungen, welche das Layer2-Transport-Protocol nutzen.

lan Native IPSec-Verbindungen über den LAN-Adapter.

lanl2tp-2k Analog wlan-l2tp-w2k

lan-l2tp-xp Analog wlan-l2tp-wxp

roadwarrior Native IPSec-Verbindungen, welche über das Internet aufgebaut werden.

rw-l2tp-2k Analog wlan-l2tp-w2k

rw-l2tp-xp Analog wlan-l2tp-wxp

Mit Hilfe dieser Verbindungsdefinitionen können sowohl Benutzer aus dem WLAN, aus dem LAN als auch von überall aus dem Internet eine abgesicherte Verbindung mittels nativem IPSec oder gekapselt im L2TP (mit dem Vorteil einer einfacheren Konfiguration unter Windows-Betriebssystemen) zum Access Point aufbauen.

[3] Vgl. The FreeRADIUS Server Project [Fre04]

[4] Natives IPSec bedeutet, dass die verschlüsselten Pakete direkt übertragen werden und in kein anderes Protokoll gekapselt sind.

In umfangreichen Tests konnte die Funktionsfähigkeit dieser Konfiguration nachgewiesen werden. Das folgende Bespiel, aufgezeichnet mit dem Programm „tcpdump“, zeigt die übertragenen Pakete eines „ICMP-Ping“ (Internet Control Message Protocol) im Falle der ungesicherten, und der mit IPSec abgesicherten Übertragung:

Ungesicherte Übertragung, WLAN Interface

```
05:05:33.369498 IP 192.168.1.178 > 192.168.1.1: icmp 40: echo request seq 1792
05:05:33.369791 IP 192.168.1.1 > 192.168.1.178: icmp 40: echo reply seq 1792
```

Im Falle der unverschlüsselten Übertragung sind sowohl die Anfrage (echo request), als auch die Antwort-Pakete (echo reply) auf der Netzwerkschnittstelle sichtbar.

Natives IPSec, WLAN Interface

```
05:11:52.539161 IP 192.168.1.178 > 192.168.1.1: ESP(spi=0xd647b1a7,seq=0x5)
05:11:52.539161 IP 192.168.1.178 > 192.168.1.1: icmp 40: echo request seq 4608
05:11:52.539161 IP 192.168.1.178 > 192.168.1.1: icmp 40: echo request seq 4608
05:11:52.540050 IP 192.168.1.1 > 192.168.1.178: ESP(spi=0x2fc1a9fe,seq=0x4)
```

Bei Verwendung der IPSec-Implementierung des Kernels sind sowohl verschlüsselte, als auch die eingehenden unverschlüsselten Datenpakete auf der Schnittstelle der drahtlosen Verbindung wahrnehmbar. Die verschlüsselten Pakete sind dabei mit ESP(...) gekennzeichnet. Ausgehende Pakete werden auf der Netzwerkschnittstelle nur verschlüsselt dargestellt - aus diesem Grund sind die Antwort-Pakete nicht im Klartext sichtbar.

IPSec über L2TP, WLAN Interface

```
05:14:08.539377 IP 192.168.1.1 > 192.168.1.178: ESP(spi=0x20c2fce9,seq=0x1d)
05:14:08.540418 IP 192.168.1.178 > 192.168.1.1: ESP(spi=0x5987a9ae,seq=0x39)
```

IPSec über L2TP, PPP Interface

```
05:15:20.698224 IP 192.168.2.200 > 192.168.2.1: icmp 40: echo request seq 1280
05:15:20.710518 IP 192.168.2.1 > 192.168.2.200: icmp 40: echo reply seq 1280
```

Nutzt man statt der nativen IPSec-Implementierung eine Kapselung der IPSec-Pakete in dem Layer2 Transport Protokoll, so wird für jede VPN-Verbindung ein PPP-Interface erstellt. Eingehende Datenpakete werden daher auf der WLAN-Schnittstelle nur verschlüsselt, also als ESP-Pakete, dargestellt. Nach der Entschlüsselung der eingehenden Pakete werden diese an die PPP-Schnittstelle weitergeleitet, verarbeitet und als Antwort über die WLAN-Schnittstelle verschlüsselt übertragen. Daher sind bei Nutzung des L2TP auf dem WLAN Interface nur verschlüsselte, auf der jeweiligen PPP-Schnittstelle nur unverschlüsselte Datenpakete sichtbar.

6.2.6 Konfiguration der Firewall

Für die verschiedenen Einsatzbereiche des IXDPG425 sind jeweils unterschiedliche Firewall-Regeln notwendig. Diese Regeln werden über die in dem Verzeichnis „/etc/shorewall“ vorhandenen Konfigurationsdateien festgelegt.

Dabei werden unterschiedliche Zonen „zones“ definiert, welche dann zu den jeweiligen Netzwerk-Schnittstellen „interfaces“ zugeordnet werden können. Über die Datei „policy“ werden globale Regeln für jede Zone definiert, die Verfeinerung dieser erfolgt dann über die Einträge in der Datei „rules“.

Mit Hilfe dieser Konfigurationsdateien sind für diese Arbeit die folgenden Firewall-Regeln definiert worden:

- Blockiere alle Ports des Access Points. Ausnahmen, beispielsweise zur Fernadministration, müssen explizit festgelegt werden.
- Verwerfe alle eingehenden Pakete aus dem Internet, die keiner Verbindung zugeordnet werden können.
- Erlaube nur ausgehende Pakete vom Access Point, die von den genutzten Diensten benötigt werden.
- Erlaube Zugriffe in das Internet aus dem WLAN und LAN.
- Erlaube Zugriffe zwischen LAN und WLAN.
- Bei Verwendung von IPSec, verwerfe alle Pakete aus dem WLAN, die nicht verschlüsselt sind.

Beim Start von Shorewall erfolgt die auf diesen Dateien basierende automatische Konfiguration des im Kernel enthaltenen Paketfilters „iptables“. Da dieser Verarbeitungsvorgang sehr langwierig ist und daher beim Start des IXDPG425 zu viel Zeit in Anspruch nimmt, wurden für die unterschiedlichen Einsatzszenarien vordefinierte Firewall-Konfigurationen abgespeichert und im Verzeichnis „/usr/lib/shorewall“ hinterlegt. Beim Bootvorgang können diese nun unter Verwendung des Programms „iptables-restore“ in den Kernel importiert werden.

6.2.7 Geschwindigkeitsanalyse

Um die Leistung der beiden Netzwerkschnittstellen bestimmen zu können, wurden verschiedene Messungen mit dem Analyseprogramm „netperf“ durchgeführt. Dabei wird ein TCP-Stream von einem Client zu dem IXDPG425 für eine festgelegte Zeitdauer übertragen und damit die durchschnittliche Übertragungsgeschwindigkeit gemessen.

Die Ergebnisse der Messungen zeigt die Abbildung 6.1.

Dabei wird deutlich, dass sowohl die LAN-, als auch die drahtlose Schnittstelle bei Verzicht auf eine Verschlüsselung mit IPSec ihre maximale Nettodatenraten erreichen. Auch

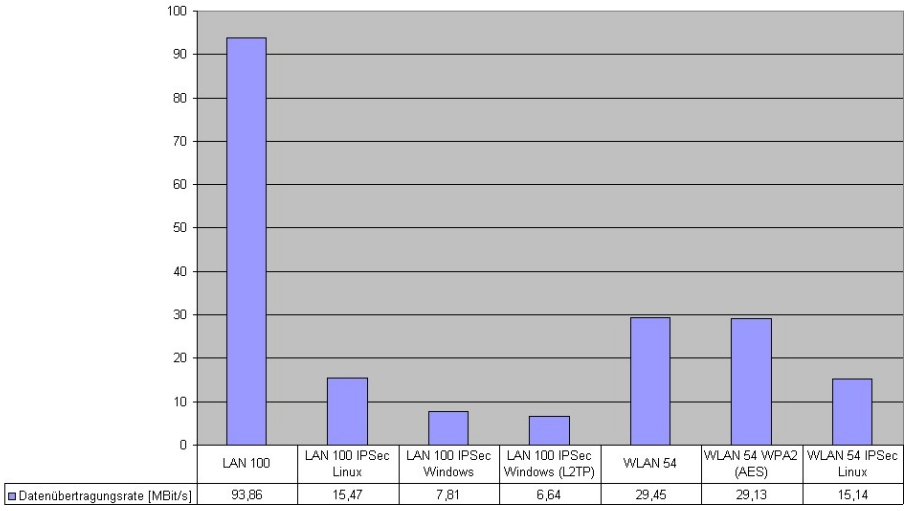

Abbildung 6.1: Ergebnisse Geschwindigkeitsmessungen IXDPG425

bei Verwendung der WLAN-Sicherheitsarchitektur WPA mit dem Verschlüsselungsalgorithmus AES ändert sich die Datenrate nicht, da sämtliche Verschlüsselungsoperationen direkt in dem WLAN-Adapter durchgeführt werden.

Nur bei Anwendung des IPSec-Protokolls sinkt die Übertragungsgeschwindigkeit deutlich auf 10 bis 15 MBit/s. Der Grund dafür liegt in der beschränkten Leistung des Prozessors IXP425 und der fehlenden Unterstützung der Beschleunigungskomponente. Würde die IPSec-Implementierung die integrierte Krypto-Beschleunigung des IXP425 nutzen können, wäre eine Steigerung der Datenraten auf 30 bis 45 MBit/s denkbar.

Beim Einsatz von Windows XP als Client-Betriebssystem sinkt die Datenübertragungsrate im Vergleich zur Linux-Implementierung. Die Gründe für diesen Geschwindigkeitsunterschied konnten leider nicht zufriedenstellend geklärt werden. Jedoch lässt sich die nochmalige Abnahme der Übertragungsgeschwindigkeit bei Nutzung des L2TP auf den zusätzlichen Paket-Overhead durch das Layer2 Transport Protokoll zurückführen.

Trotzdem kann die Leistung des IXDPG425 als WLAN Access Point und Router als völlig ausreichend bewertet werden, solange das IPSec-Protokoll hinreichend sparsam angewendet wird. Verwendet man statt dessen IPSec grundsätzlich zur Verschlüsselung des WLAN-Verkehrs und ist der Austausch großer Datenmengen innerhalb des drahtlosen Netzes notwendig, so wird die Geschwindigkeit der Datenübertragung drastisch verringert.

6.3 Konfiguration der Clients

6.3.1 IPSec mit Linux

Bei Verwendung des Betriebssystems Linux auf den Endgeräten der Benutzer können die verschiedenen, unter Linux erhältlichen IPSec-Implementierungen verwendet werden. Am einfachsten gestaltet sich dabei die Nutzung des auf dem Access Point eingesetzten Paketes „Openswan“, da in diesem Fall die Konfigurationsdatei dessen übernommen werden kann.

6.3.2 Natives IPSec mit Windows 2000/XP

Die aktuellen Version des Microsoft Windows Betriebssystems enthalten bereits eine Unterstützung des IPSec-Protokolls. Diese kann jedoch nur sehr aufwändig durch die „IP-Sicherheitsrichtlinienverwaltung“ der Microsoft Management-Konsole angepasst werden. Daher empfiehlt sich die Verwendung eines speziellen VPN-Clients, welcher zumeist nur kommerziell erworben werden kann.

Eine kostenfreie Implementierung bietet das Hilfsprogramm VPN-Tool[5] von Marcus Müller. Dieses benötigt eine Konfigurationsdatei, welche syntaktisch ähnlich zu der von Openswan verwendeten ist. Eine Beispielkonfiguration für den Zugriff auf den mit IPSec abgesichterten Access Point lässt sich folgendermaßen realisieren:

Listing 6.4: Konfiguration VPN Tool

```
conn lan
  left=192.168.1.1
  leftsubnet=*
  rightca="C=DE, S=Sachsen, L=Dresden, O=Testfirma, OU=WLAN EAP,
           CN=WLAN CA, E=ca@test.de"
  right=%any
  network=lan
  auto=start
  pfs=yes
```

Nach dem Start des VPN Tools wird diese Konfiguration eingelesen und damit eine „IP-Sicherheitsrichtlinie“ für Windows erstellt.

6.3.3 IPSec über L2TP

Zum Aufbau einer IPSec-Verbindung ist, wie im vorangegangenen Abschnitt gezeigt, die Exisitenz eines zusätzlichen Programmes auf dem Client notwendig. Jedoch ist dies nicht immer möglich, weshalb nach einer alternativen Variante zur Verbindungskonfiguration gesucht wurde. Eine Lösung stellt hierbei die Kapselung des IPSec-Protokolls in das von

[5] Siehe Homepage VPN-Tool [Mül03]

Windows komfortabel unterstützte Layer2 Transport Protokoll dar. Hierbei kann die Konfiguration des Tunnels durch den in Windows integrierten „Assistent für neue Verbindungen“ durchführt werden. Für diese Art der Verbindung ist die Existenz eines L2TP-Dämon auf dem Access Point notwendig - dieser wurde durch das Paket „rp-l2tp“ auf diesem installiert.

Eine mit IPSec über L2TP abgesichterte Verbindung kann in Windows XP wie folgt konfiguriert werden:

- Assistent für neue Verbindungen starten.
- Auswahl: Verbindung mit dem Netzwerk am Arbeitsplatz herstellen.
- Auswahl: VPN-Verbindung.
- Festlegen eines Namens für die VPN-Verbindung.
- Auswahl: Keine Anfangsverbindung automatisch wählen.
- IP-Adresse des Access Points eingeben (Bsp: 192.168.1.1).
- Auswahl: Fertigstellen.
- Im Fenster „Verbindung mit ...“ die Schaltfläche „Eigenschaften“ klicken.
- Im Reiter „Netzwerk“ den VPN-Typ auf „L2TP-IPSec-VPN“ einstellen.
- Das Eigenschaften-Fenster schliessen.
- Unter Verwendung der im Access Point konfigurierten Benutzername/Passwort-Kombination die Verbindung aktivieren.

Der Vorteil dieser Art der Verbindungskonfiguration liegt in der Tatsache, dass keine zusätzliche Software benötigt wird. Jedoch wirkt sich der Overhead, hervorgerufen durch die Kapselung der IPSec-Pakete in das Layer2 Transport Protokoll, negativ auf die Geschwindigkeit der Datenübertragung aus (Vgl. Kapitel 6.2.7).

Kapitel 7

Validierung

Nachdem im vorangegangen Kapitel die einzelnen Softwaremodule bezüglich ihrer Funktionalität getestet wurden, wird nun die Gesamtfunktionalität des Access Points unter Einbeziehung der im Kapitel 5.2 vorgestellten Szenarien untersucht.

7.1 Testumgebung

Für den Test der verschiedenen Szenarien wurde das Netzwerkgateway IXDPG425 mit der im Kapitel 6 erstellten Embedded Linux Distribution eingesetzt. Als Clients kamen dabei zwei PCs mit folgender Konfiguration zum Einsatz:

- Linux-Betriebssystem
  - SUSE-Distribution 9.2 mit Kernel 2.6.8.2
  - IPSec-Implementierung: Openswan 2.3.0
  - WLAN-Treiber: MADWIFI Version 0.9.6.0
  - WPA-Supplicant: wpa_supplicant Version 0.3.9
- Windows-Betriebssystem
  - Microsoft Windows XP Service Pack 2
  - WLAN-Treiber: Atheros-Treiber Version 4.1.2.56 (enthält WPA-Supplicant)
  - IPSec-Implementierung: in Windows XP enthalten
  - IPSec-Konfiguration: VPN Tool ([Mül03])

Beide Clients verfügen über eine WLAN-Verbindung zum Access Point. Auf beiden Clients sind zudem die für die Authentifikation notwendigen Zertifikate installiert. Zum Test der Verbindung in das leitungsgebundene Netzwerk wurde ein Linux-Server mit der LAN-Schnittstelle des IXDPG425 verbunden.

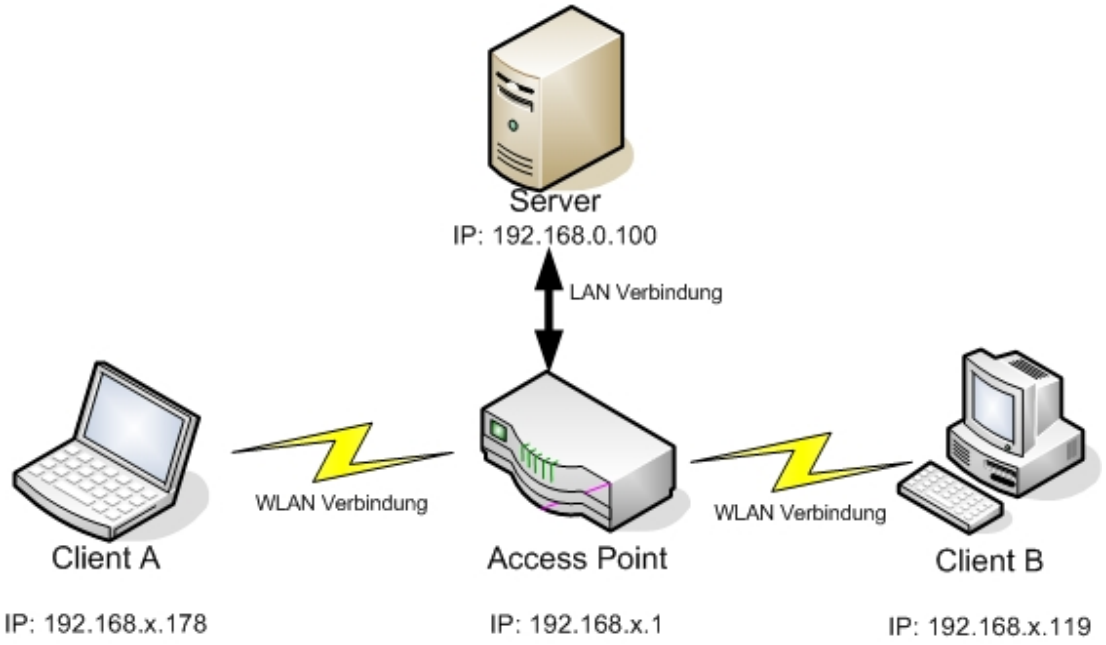

Abbildung 7.1: Testumgebung

7.2 WLAN Access Point ohne Sicherheitsfunktionen

Im ersten Szenario stellt der Access Point nur die reine WLAN-Konnektivität zur Verfügung. Dabei wird die drahtlose Schnittstelle des IXPDG425 (ath0) mit der LAN-Schnittstelle (eth0) über die im Kernel enthaltene Bridge-Funktionalität verbunden. Dies erfolgt über die folgenden Befehle:

```
brctl addbr br0
brctl addif br0 ath0
brctl addif br0 eth0
ifconfig ath0 0.0.0.0
ifconfig eth0 0.0.0.0
ifconfig br0 192.168.0.1
```

Der Access Point erhält damit die IP-Adresse 192.168.0.1, die Clients müssen mangels DHCP-Server manuell konfiguriert werden und erhalten die IP-Adresse 192.168.0.119 bzw. 192.168.0.178.

Nach Abschluß der Konfiguration konnten erfolgreich Verbindungen zum Server im LAN, aber auch zwischen den beiden Clients aufgebaut werden.

ICMP-Ping von Client A zum Server

```
16:49:31.159893 IP 192.168.0.178 > 192.168.0.100: icmp 64: echo request seq 2
16:49:31.161091 IP 192.168.0.100 > 192.168.0.178: icmp 64: echo reply seq 2
```

ICMP-Ping von Client A zum Client B

```
16:52:12.807062 IP 192.168.0.178 > 192.168.0.119: icmp 64: echo request seq 1
16:52:12.807119 IP 192.168.0.119 > 192.168.0.178: icmp 64: echo reply seq 1
```

7.3 WLAN Access Point mit IPSec-Sicherheit

Zur Absicherung der WLAN-Verbindung wird im zweiten Szenario das IPSec-Protokoll eingesetzt. Das folgende Protokoll zeigt einen Verbindungsaufbau zwischen Client A und dem Access Point:

```
No.  Source          Destination     Protocol Info
  1  192.168.0.178   192.168.0.1     ISAKMP   Identity Protection (Main Mode)
  Payload: Security Association
  2  192.168.0.1     192.168.0.178   ISAKMP   Identity Protection (Main Mode)
  Payload: Security Association
  3  192.168.0.178   192.168.0.1     ISAKMP   Identity Protection (Main Mode)
  Payload: Key Exchange
  4  192.168.0.1     192.168.0.178   ISAKMP   Identity Protection (Main Mode)
  Payload: Key Exchange
  5  192.168.0.178   192.168.0.1     ISAKMP   Identity Protection (Main Mode)
  Payload: Identification
  6  192.168.0.1     192.168.0.178   ISAKMP   Identity Protection (Main Mode)
  Payload: Identification
  7  192.168.0.178   192.168.0.1     ISAKMP   Quick Mode, Payload: Hash
  8  192.168.0.1     192.168.0.178   ISAKMP   Quick Mode, Payload: Hash
  9  192.168.0.178   192.168.0.1     ISAKMP   Quick Mode, Payload: Hash
```

Nach den verschiedenen Phasen des ISAKMP-Austausches ist die IPSec-Verbindung aufgebaut. Danach werden sämtliche Pakete zwischen Client A und Access Point verschlüsselt mit dem ESP-Protokoll übertragen.

7.4 WLAN Access Point mit IEEE 802.11i

Wird die WLAN-Verbindung mit der in der Norm IEEE 802.11i definierten Sicherheitsarchitektur abgesichert, so muss bei jedem Verbindungsaufbau eine Authentifizierung mit dem EAP-Protokoll durchgeführt werden. Beim Einsatz von Linux auf dem PC des Clients muss dafür die Software „wpa_supplicant“ benutzt werden, unter Windows ist diese Funktionalität Bestandteil des verwendeten WLAN-Treibers.

Einen erfolgreichen Verbindungsaufbau zwischen Client A und Access Point zeigt das folgende Protokoll:

```
No  Source          Destination     Protocol Info
1   192.168.0.178   192.168.0.1     EAPOL    Start
2   192.168.0.1     192.168.0.178   EAP      Request, Identity [RFC3748]
3   192.168.0.178   192.168.0.1     EAP      Response, Identity [RFC3748]
4   192.168.0.1     192.168.0.178   EAP      Request, Identity [RFC3748]
5   192.168.0.178   192.168.0.1     EAP      Response, Identity [RFC3748]
6   192.168.0.1     192.168.0.178   EAP      Request, EAP-TLS [RFC2716] [Aboba]
7   192.168.0.178   192.168.0.1     TLS      Client Hello
8   192.168.0.1     192.168.0.178   TLS      Server Hello, Certificate,
    Certificate Request, Server Hello Done
9   192.168.0.178   192.168.0.1     EAP      Response, EAP-TLS [RFC2716] [Aboba]
10  192.168.0.1     192.168.0.178   TLS      Server Hello, Certificate,
    Certificate Request, Server Hello Done
11  192.168.0.178   192.168.0.1     TLS      Certificate, Client Key Exchange,
    Certificate Verify, Change Cipher Spec, Encrypted Handshake Message
12  192.168.0.1     192.168.0.178   EAP      Request, EAP-TLS [RFC2716] [Aboba]
13  192.168.0.178   192.168.0.1     TLS      Certificate, Client Key Exchange,
    Certificate Verify, Change Cipher Spec, Encrypted Handshake Message
14  192.168.0.1     192.168.0.178   TLS      Change Cipher Spec,
    Encrypted Handshake Message
15  192.168.0.178   192.168.0.1     EAP      Response, EAP-TLS [RFC2716] [Aboba]
16  192.168.0.1     192.168.0.178   EAP      Success
17  192.168.0.1     192.168.0.178   EAPOL    Key
18  192.168.0.178   192.168.0.1     EAPOL    Key
19  192.168.0.1     192.168.0.178   EAPOL    Key
20  192.168.0.178   192.168.0.1     EAPOL    Key
21  192.168.0.1     192.168.0.178   EAPOL    Key
22  192.168.0.178   192.168.0.1     EAPOL    Key
```

Den Begin des Verbindungsaufbaus kennzeichnet das Paket „EAPOL Start“. Danach wird über das EAP-Protokoll das Verfahren EAP-TLS ausgehandelt und mit diesem eine auf Client- und Server-Zertifikaten basierende Authentifizierung durchgeführt. Nach dem erfolgreichen Authentifizierungsvorgang (EAP Success) findet der Schlüsselaustausch über das EAPOL-Protokoll statt.

7.5 WLAN Router

Bei der Erweiterung des WLAN Access Points um die Routing-Funktionalität kann dieser auch als Schnittstelle in das Internet verwendet werden. Ein an die LAN-Schnittstelle angeschlossenes DSL-Modem stellt über das PPPoE-Protokoll eine Internet-Verbindung her, die dann durch die Benutzer im WLAN und im LAN genutzt werden kann.

Die beiden Schnittstellen WLAN und LAN werden hierbei nicht mehr über eine Bridge miteinander verbunden. Statt dessen werden beiden Schnittstellen unterschiedliche IP-Adressbereiche zugeordnet - die LAN-Schnittstelle nutzt das Netz 192.168.0.0/24, den Benutzern des Wireless LAN werden IP-Adressen aus dem Netz 192.168.1.0/24 zugeordnet.

Aufgrund der Integration des Pakets „dnsmasq“ erfolgt in diesem Szenario die Konfiguration der Clients über das DHCP-Protokoll. Dabei werden automatisch Parameter wie IP-Adresse, Standard-Gateway und DNS-Server zum Client übertragen.

Verbindet sich der Client A mit dem WLAN Router, wird zunächst eine Authentifizierung mit WPA durchgeführt. Im Anschluß daran fordert der Client seine Konfiguration über das DHCP-Protokoll an. Der Client A erhält dabei die IP-Adresse 192.168.1.178 und als Standard-Gateway und DNS-Server die IP-Adresse des Routers.

Nun kann der Client die durch den Router hergestellte Internet-Verbindung nutzen. Das folgende Beispiel zeigt die auf der WLAN-Schnittstelle des Routers übertragenen Datenpakete, um einen ICMP-Ping vom Client A zum Ziel „wwww.heise.de“ durchzuführen.

```
03:31:46  IP 192.168.1.178.1024 > 192.168.1.1.domain:  39168+ A? heise.de. (26)
03:31:46  IP 192.168.1.1.domain > 192.168.1.178.1024:  39168 1/0/0 (42)
03:31:46  IP 192.168.1.178 > redirector.heise.de: icmp 64: echo request seq 1
03:31:46  IP redirector.heise.de > 192.168.1.178: icmp 64: echo reply seq 1
```

Zu Beginn wird eine DNS-Anfrage an den von „dnsmasq“ implementierten DNS-Forwarder gestellt. Dieser sendet die IP-Adresse des gewünschten Servers an den Client, womit dieser nun die Ping-Anfrage direkt absenden kann.

7.6 WLAN Router mit unterschiedlichen Benutzergruppen

Für die Umsetzung des letzten Szenarios wurde der Hostap-Dämon angepasst, um basierend auf zur WPA-Authentifizierung genutzten Benutzernamen eine dynamische Anpassung der Firewall durchzuführen (vgl. Kapitel 6.1.5). Dabei wird für jede WLAN-Assoziierung, die sich mit dem Benutzernamen „gast“ anmeldet, eine Firewallregel definiert, die den Zugriff auf den LAN-Adressbereich 192.168.0.0/24 verbietet.

Meldet sich der WLAN-Client mit einem nichtreglementierten Benutzernamen an, ist ein ICMP-Ping an die LAN-Schnittstelle des Routers möglich:

```
client:~# ping 192.168.0.1
PING 192.168.0.1 (192.168.0.1) 56(84) bytes of data.
64 bytes from 192.168.0.1: icmp_seq=1 ttl=64 time=1.76 ms
64 bytes from 192.168.0.1: icmp_seq=2 ttl=64 time=0.99 ms
```

Wird nun statt dessen der Benutzername „gast“ für die WPA-Authentifizierung verwendet, werden sämtliche Pakete in das Netz 192.168.0.0/24 zurückgewiesen:

```
client:~# ping 192.168.0.1
PING 192.168.0.1 (192.168.0.1) 56(84) bytes of data.
From 192.168.0.1: icmp_seq=1 Destination Port Unreachable
From 192.168.0.1: icmp_seq=2 Destination Port Unreachable
```

Somit kann kein als Gast angemeldeter Benutzer Zugriff auf die im LAN erreichbaren Netzwerkgeräte erhalten.

Kapitel 8

Fazit und Perspektiven

8.1 Zusammenfassung

Mit der im Verlauf dieser Diplomarbeit entstandenen Embedded Linux Distribution für WLAN Access Points lassen sich unterschiedliche Wireless LAN Einsatzszenarien unter Verwendung des Intel IXDPG425 realisieren. Dabei stehen neben den aktuellen Sicherheitsprotokollen wie WPA, WPA2 und IPSec zur Absicherung der drahtlosen Verbindung weitere Softwaremodule zum Aufbau eines komplexen WLAN Routers zur Verfügung.

Zur Realisierung dieser Distribution mussten eine Vielzahl von möglichen Open Source Projekten auf ihre Eignung und Funktionsfähigkeit untersucht und anschließend für die Verwendung auf dem IXDPG425 konfiguriert werden. Häufig treten dabei Fehler und Probleme auf, welche nur durch den direkten Kontakt zu den Entwicklern und durch zeitlich aufwendige Funktionstests gelöst werden können.

Somit können die folgenden, in der Aufgabenstellung festgelegten Funktionen realisiert werden:

WLAN Funktionalität Der WLAN Treiber „MADWIFI“ wurde erfolgreich für die Verwendung auf dem IXDPG425 kompiliert. Trotz des frühen Entwicklungsstadiums sind in der aktuellsten Version keine Einschränkungen bezüglich Funktionalität und Geschwindigkeit nachgewiesen worden.

Firewall Die im Kernel integrierte Firewall schützt den WLAN Access Point auf allen Netzwerkschnittstellen. Zur Vereinfachung der Konfiguration wurde die Software „Shorewall“ mit positivem Ergebnis validiert. Dafür wurde eine Beispielkonfiguration erstellt, die für die verschiedenen Netzwerkschnittstellen adäquate Firewall-Regeln definiert.

VPN Der Access Point verfügt mit dem Paket „OpenSwan“ über eine Software zum Aufbau „Virtueller Privater Netzwerke“. Mit den dafür erstellten Konfigurationsdateien konnten erfolgreich VPN-Verbindungen sowohl über das Internet, als auch über die drahtlose sowie die leitungsgebundene Netzwerkschnittstelle aufgebaut werden.

Authentifizierung Sämtliche, auf dem IXDPG425 eingesetzten Sicherheitsarchitekturen verwenden aktuelle und sichere Verfahren zur Authentifizierung der Benutzer. Diese

können sowohl auf Zertifikaten als auch auf Benutzernamen und Passwörtern basieren.

Accounting Bei der Benutzung eines externen RADIUS-Servers können Zugriffsinformationen zur Abrechnung gespeichert und ausgewertet werden.

Unterstützung mehrerer Benutzergruppen Mit Hilfe der vom Autor vorgenommenen Anpassung des Hostap-Dämon kann das Verhalten der Firewall vom dem zur Authentifizierung verwendeten Benutzernamen abhängig gemacht werden. Damit lassen sich unterschiedliche Zugriffsrechte für verschiedene Benutzergruppen rudimentär implementieren.

8.2 Mögliche Erweiterungen

Der große Vorteil der realisierten Distribution liegt in der konsequenten Verwendung quelloffener Software. Somit lässt sich der WLAN Access Point relativ einfach um neue Funktionalitäten erweitern - beispielsweise durch Einfügen neuer Softwaremodule oder durch Aktualisierung der vorhandenen Programme.

Beispiele für zukünftige Erweiterungen:

- Einbau eines zusätzlichen WLAN-Adapters zur Überwachung der Funkkanäle und zur Erkennung böswilliger WLAN-Clients bzw. Access Points
- Anschluss von USB-Festplatten zur gemeinsamen Nutzung im Netzwerk
- Integration der Telefonschnittstellen des IXDPG425 für VoIP-Verbindungen
- Installation einer USB-Kamera zur Raumüberwachung

Neben der Erweiterung des Funktionsumfangs könnte die Arbeit hinsichtlich der Beschleunigung kryptographischer Operationen fortgeführt werden. Dafür müsste entweder selbständig die IPSec-Implementation des Kernels 2.6 auf die Verwendung der Intel Access Library angepasst, oder auf eine Erweiterung von IPSec zur Nutzung des „OpenBSD Cryptographic Framework“ gewartet werden. Aufgrund des begrenzten Zeitrahmens dieser Diplomarbeit konnte dieses Ziel leider nicht verfolgt werden.

Die für eine kommerzielle Vermarktung notwendige Existenz einer komfortablen Konfigurationsumgebung, beispielsweise per Web-Schnittstelle, konnte aus Zeit- und Komplexitätsgründen ebenfalls nicht in dieser Diplomarbeit berücksichtigt werden.

Somit wird die Funktionalität der im Kapitel 5.1 vorgestellten kommerziellen WLAN Access Points nicht vollständig erreicht. Dies war allerdings auch nicht Ziel dieser Arbeit. Für zusätzliche Funktionen wie die komfortable Verwaltung unterschiedlicher Benutzergruppen im WLAN sowie die Konfiguration sämtlicher Software-Module über eine Web-Schnittstelle wäre ein wesentlich höherer Entwicklungsaufwand notwendig.

8.3 Bewertung der Lösung

Im Vergleich zu kommerziellen WLAN Distributionen existieren neben der Erweiterbarkeit jedoch noch weitere Vorteile bei der Nutzung der vorliegenden Open Source Distribution:

Offenheit Nahezu alle Module liegen im Quelltext vor. Dadurch können Kontrollen bezüglich Fehler und unerwünschter Funktionen erfolgen.

Lizenzfreiheit Alle in dieser Arbeit verwendeten Module können kostenfrei genutzt werden. Es sind keine Lizenzgebühren fällig.

Anpassbarkeit Der Großteil der genutzten Programme untersteht der GPL und kann daher im Quellcode verändert werden. Somit können rechtlich unbedenklich Funktionserweiterungen und Anpassungen der Software durchgeführt werden.

Sicherheit Auftretende Sicherheitslücken werden von der Open Source-Gemeinde häufig sehr zeitnah behoben. Es entfällt daher das Warten auf ein Update des Hardware-Herstellers.

Konfigurationsfreiheit Viele Open Source-Programme erlauben einen wesentlich größeren Konfigurationsspielraum als kommerzielle Produkte. Die Flexibilität der Distribution erhöht sich, damit aber auch zugleich die Anfälligkeit gegenüber Fehlkonfigurationen.

Interoperabilität Während kommerzielle Produkte häufig proprietäre Protokolle und Dienste nutzen, wird bei Open Source Projekten mehr Wert auf die geltenden Standards gelegt. Daher erhöht sich die Kompatibilität unterschiedlicher Geräte und Dienste.

Vernachlässigt werden dürfen hierbei aber auch nicht die Nachteile der OpenSource-Implementierung. Aufgrund der fehlenden Anwenderunterstützung und der häufig nicht vorhandenen Dokumentation kann sich die Inbetriebnahme einer OpenSource-Software teilweise sehr schwierig gestalten. Zwar existieren für fast jedes OpenSource-Projekt Mailinglisten zur Anwenderunterstützung - aber auch mit deren Hilfe können nicht alle auftretenden Probleme vollständig beseitigt werden.

Zusammenfassend lässt sich feststellen, dass kommerzielle Produkte bezüglich Komfort und herstellerspezifischen Erweiterungen deutliche Vorteile ausweisen, während Lösungen basierend auf quelloffener Software häufig höheres Erweiterungspotenzial besitzen sowie höheren Sicherheitsanforderungen genügen.[1]

Die vorliegenden Open Source WLAN-Distribution ermöglicht den Aufbau eines WLAN Access Points bzw. WLAN Routers mit sämtlichen in der Aufgabenstellung geforderten Funktionen. Vor allem aufgrund der unkomfortablen Konfiguration eignet sich das Ergebnis weniger zum Einsatz in Produktivumgebungen, kann jedoch sehr gut als Grundlage

[1] Zur Problematik kommerzieller Sicherheitssoftware sei auf den Artikel „Weltpolitik beeinflusst die Internet-Sicherheit“, erschienen in der Computer Zeitung, Ausgabe 27/2005, verwiesen. [Sch05]

zum Aufbau eines solchen Systems genutzt werden. Zukünftige Entwicklungen, beispielsweise die Erweiterung des WLAN-Treibers um Multi-SSID-Funktionalität, könnten in absehbarer Zeit den Funktionsumfang und die Leistungsfähigkeit der vorgestellten Lösung zusätzlich erhöhen.

Anhang A

Konfigurationsdateien

A.1 Dnsmasq

Listing A.1: dnsmasq.conf

```
# Konfigurationsdatei für dnsmasq.
#
# Leite nur vollständige Domain-Namen weiter.
domain-needed
# Keine Adressen aus privaten Adress-Bereichen weiterleiten.
bogus-priv


# Systembedingte DNS-Anfragen von Windows 2000 filtern.
filterwin2k

# Datei, in der die zu verwendenden Name-Server zu finden sind.
resolv-file=/var/run/ppp/resolv.conf

# Zu verwendende Netzwerk-Schnittstellen
interface=eth0
interface=ath0

# Ports an Netzwerk-Schnittstellen binden.
bind-interfaces

# Domainname im internen Netzwerk
domain=test.org

# IP-Adress-Bereich für DHCP-Server
# dhcp-range=bezeichnung,startadresse,endadresse,lease-zeitdauer
dhcp-range=lan,192.168.0.100,192.168.0.200,2h
dhcp-range=wlan,192.168.1.100,192.168.1.200,2h

# DHCP-Optionen fuer LAN
```

```
31 # Standard-Gateway
32 dhcp-option=lan,6,192.168.1.1
33 # DNS-Server
34 dhcp-option=lan,3,192.168.1.1
35
36 # DHCP-Options fuer WLAN
37 # Standard-Gateway
38 dhcp-option=wlan,6,192.168.1.1
39 # DNS-Server
40 dhcp-option=wlan,3,192.168.1.1
```

A.2 IPSec

Listing A.2: ipsec.conf

```
1  version 2.0     # Version 2 der IPSec-Spezifikation verwenden
2
3  # Grundkonfiguration
4  config setup
5   # Debugausgaben festlegen
6   klipsdebug="control"
7   plutodebug="control"
8
9  # Standard-Konfiguration für eine Verbindung
10 conn %default
11         # Automatisch beim Start hinzufügen
12         auto=add
13         # Authentifikation durch RSA-Signatur
14         authby=rsasig
15         # Für den rechten Partner jede IP-Adresse zulassen
16         right=%any
17         # Zertifikatsidentifizierung
18         rightid='C=DE, ST=Sachsen, L=Dresden, O=Testfirma,
19                  OU=WLAN EAP, CN=*, E=*'
20         # Dateiname für eigenes Zertifikat
21         leftcert=cert-srv.pem
22         # Anzahl der Versuche, einen Schlüssel auszutauschen
23         keyingtries=0
24
25 # Verbindung für verschlüsselte WLAN-Übertragung
26 conn wlan
27         # Eigene IP-Adresse
28         left=192.168.1.1
29         # Sämtliche Pakete sollen verschlüsselt werden.
```

```
30          leftsubnet=0.0.0.0/0
31          # Perfect Forward Secrecy aktivieren
32          pfs=yes
33
34
35  # Verbindung für verschlüsselte WLAN-Übertragung mit L2TP
36  # unter Windows XP
37  conn wlan-l2tp-xp
38          left=192.168.1.1
39          # Port für L2TP-Anfragen
40          leftprotoport=17/1701
41          rightprotoport=17/1701
42          # Transport-Modus verwenden
43          type=transport
44          # Perfect Forward Secrecy deaktivieren
45          pfs=no
46
47  # Verbindung für verschlüsselte WLAN-Übertragung mit L2TP
48  # unter Windows 2000
49  conn wlan-l2tp-2k
50          left=192.168.1.1
51          leftprotoport=17/0
52          rightprotoport=17/1701
53          type=transport
54          pfs=no
55
56  # Verbindung für verschlüsselte LAN-Übertragung
57  conn lan
58          # Eigene IP-Adresse
59          left=192.168.0.25
60          # Sämtliche Pakete sollen verschlüsselt werden.
61          leftsubnet=0.0.0.0/0
62          # Perfect Forward Secrecy aktivieren
63          pfs=yes
64
65  conn lan-l2tp-xp
66          left=192.168.0.25
67          leftprotoport=17/1701
68          rightprotoport=17/1701
69          type=transport
70          pfs=no
71
72  conn lan-l2tp-2k
73          left=192.168.0.25
74          leftprotoport=17/0
```

```
75          rightprotoport=17/1701
76          type=transport
77          pfs=no
78
79  # Verbindung für IPSec-Verbindungen aus dem Internet, right=%any
80  conn roadwarrior-net
81          leftsubnet=192.168.0.0/16
82          also=roadwarrior
83
84  conn roadwarrior
85          left=%defaultroute
86          pfs=yes
87
88  conn rw-l2tp-xp
89          left=%defaultroute
90          leftprotoport=17/1701
91          rightprotoport=17/1701
92          type=transport
93          pfs=no
94
95  conn rw-l2tp-2k
96          left=%defaultroute
97          leftprotoport=17/0
98          rightprotoport=17/1701
99          type=transport
100         pfs=no
101
102 include /etc/ipsec/ipsec.d/examples/no_oe.conf
```

A.3 L2TP

Listing A.3: l2tp.conf

```
1
2  # Globale Einstellungen
3  global
4
5  # Zusätzliche PPP-Module laden
6  load-handler "sync-pppd.so"
7  load-handler "cmd.so"
8
9  # An Port 1701 binden
10 listen-port 1701
11
```

```
12 # Synchrones PPP konfigurieren
13 section sync-pppd
14
15 # PPP-Interface mit Process-ID bezeichnen.
16 set-ppp-if-name 1
17
18 # Importiere PPP-Optionen aus Datei options.lns
19 lns-pppd-opts "file /etc/ppp/options.lns"
20
21 # Partner-Konfiguration
22 section peer
23
24 # Verbindungspartner aus allen IP-Adress-Bereichen erlaubt
25 peer 0.0.0.0
26 mask 0
27 port 1701
28
29 # Synchrones PPP verwenden
30 lac-handler sync-pppd
31 lns-handler sync-pppd
32 hide-avps no
```

A.4 PPP

Listing A.4: options.lns

```
192.168.2.1:
debug
nopcomp
noaccomp
require-chap
refuse-pap
lcp-echo-interval 30
lcp-echo-failure 6
ms-dns 217.237.151.97
noccp auth crtscts
idle 1800
mtu 1400
mru 1400
nodefaultroute
lock
proxyarp
connect-delay 5000

plugin dhcpc.so
dhcp-interface lo
dhcp-relay-address 192.168.1.1
dhcp-server 192.168.1.1
dhcp-subnet-selection 192.168.2.1
```

A.5 Hostap-Dämon

Listing A.5: hostapd-intradius.conf

```
# Zu verwendende WLAN Schnittstelle
interface=ath0

# Treiber für WLAN Schnittstelle
driver=madwifi

# Debug-Parameter
logger_syslog=-1
logger_syslog_level=2
logger_stdout=-1
logger_stdout_level=2
debug=0

```

```
14 # Datei für Statusinformationen
15 dump_file=/tmp/hostapd.dump
16
17 # Schnittstelle für externe Programme
18 ctrl_interface=/var/run/hostapd
19 ctrl_interface_group=0
20
21
22 ##### IEEE 802.11 Konfiguration ############################################
23
24 # Zu verwendende SSID
25 ssid=test-intradius
26
27 # Authentifikationsalgorithmus auswählen
28 auth_algs=3
29
30 ##### IEEE 802.1X-REV Konfiguration ########################################
31
32 # IEEE 802.1X aktivieren
33 ieee8021x=1
34
35 ##### Integrierter EAP Authentikator ########################################
36
37 # Benutze internen EAP-Server
38 eap_server=1
39
40 # Datei für EAP-Benutzerdatenbank
41 eap_user_file=/etc/hostapd/hostapd.eap_user
42
43 # CA Zertifikat (PEM oder DER file) für EAP-TLS/PEAP/TTLS
44 ca_cert=/etc/hostapd/cacert-crl.pem
45
46 # Server Zertifikat (PEM or DER file) für EAP-TLS/PEAP/TTLS
47 server_cert=/etc/hostapd/cert-srv.pem
48
49 # Privater Schlüssel für Server.
50 private_key=/etc/hostapd/key-srv.pem
51
52 # Passwort für Server-Schlüssel
53 private_key_passwd=password
54
55 # Aktiviere die Prüfung auf zurückgezogene Zertifikate
56 check_crl=1
57
58
59 ##### WPA/IEEE 802.11i Konfiguration ########################################
```

```
60
61 # Aktiviere WPA
62 wpa=2
63
64 # Schlüsselmanagement mit EAP
65 wpa_key_mgmt=WPA-EAP
66
67 # Auswahl der Verschlüsselungsalgorithmen
68 wpa_pairwise=TKIP CCMP
69
70 # Zeitinterval zum Wechseln des Gruppenschlüssels
71 wpa_group_rekey=600
72
73 # Wechsel Schlüssel beim Verlassen eines Clients
74 wpa_strict_rekey=1
75
76 # Zeitinterval zum Wechseln des Hauptschlüssels
77 wpa_gmk_rekey=86400
```

Bei Existenz eines externen RADIUS-Servers kann auch dieser zur Authentifikation und zur Abrechnung verwendet werden. Dabei wird der interne EAP-Server deaktiviert und es müssen die Parameter zur Kommunikation mit dem RADIUS-Server eingestellt werden.

Listing A.6: Ausschnitt hostapd-radius.conf

```
1 ###### IEEE 802.1X-REV Konfiguration ###################################
2
3 # IEEE 802.1X aktivieren
4 ieee8021x=1
5
6 ###### Integrierter EAP Authentikator ###################################
7
8 # Deaktiviere internen EAP-Server
9 eap_server=0
10
11 ###### RADIUS Konfiguration ##############################################
12
13 # Eigene IP-Adresse für RADIUS-Kommunikation
14 own_ip_addr=192.168.0.30
15
16 # NAS-Identifizierung
17 nas_identifier=test
18
19 # RADIUS Authentication Server
20
21 # IP-Adresse des RADIUS-Server
22 auth_server_addr=192.168.0.1
```

```
23 # Port-Nummer
24 auth_server_port=1820
25 # Gemeinsames Geheimnis zur Kommunikation
26 auth_server_shared_secret=test
27
28 # RADIUS Accounting Server
29 acct_server_addr=192.168.0.1
30 acct_server_port=1821
31 acct_server_shared_secret=test
```

Listing A.7: Benutzerdatenbank hostapd.eap_user

```
1  # Phase 1
2
3  # Benutzername für EAP-TLS, kein Password
4  "test-tls"       TLS
5
6  # Anonymes TTLS und PEAP in Phase 1
7  *                TTLS,PEAP
8
9  # Phase 2 (Getunnelt in EAP-PEAP oder EAP-TTLS)
10
11 # Benutzername und Passwort für MSCHAPV2-Authentifikation
12 "test-mschap"    MSCHAPV2         "password"       [2]
```

A.6 Shorewall

Listing A.8: Schnittstellen-Konfiguration „interfaces“

```
#ZONE    INTERFACE       BROADCAST       OPTIONS
GATEWAY

# WLAN Schnittstelle
wlan     ath0            detect          dhcp

# erste LAN Schnittstelle
lan      eth0            detect

# zweite LAN Schnittstelle
lan2     eth1            detect

# Internet Schnittstelle (PPP Einwahl)
net      ppp0            -

# VPN Schnittstelle (bei Verwendung von L2TP)
vpn      ppp+            -
```

Listing A.9: Netzwerkzonen-Konfiguration „zones“

```
#ZONE    DISPLAY         COMMENTS
net      Net             Internet
lan      Lan             Lokales Netzwerk eth0
lan2     Lan2            Lokales Netzwerk eth1
wlan     WLan            Drahtloses Netzwerk
vpn      VPN             Virtual Private Network
```

Listing A.10: Konfiguration der Policy „policy“

```
#SOURCE         DEST            POLICY          LOG
#                                               LEVEL

# Vom LAN ins Internet erlauben
lan             net             ACCEPT

# Nur wenn IPSEC aktiv, Zugriffe ins Internet erlaubt
vpn             net             ACCEPT

# Zugriffe ins Internet für alle WLAN-Geräte erlauben
wlan            net             ACCEPT

# Vom LAN ins WLAN erlauben
lan             wlan            ACCEPT
```

```
15 # Vom LAN ins VPN erlauben
16 lan             vpn             ACCEPT
17 # Vom WLAN ins LAN erlauben
18 wlan            lan             ACCEPT
19 # Vom WLAN ins VPN erlauben
20 wlan            vpn             ACCEPT
21 # Vom VPN ins LAN erlauben
22 vpn             lan             ACCEPT
23 # Vom VPN ins WLAN erlauben
24 vpn             wlan            ACCEPT
25
26 # Eingehende Pakete verwerfen
27 net             all             DROP            info
28
29 # Alle anderen Pakete zurückweisen
30 all             all             REJECT          info
```

Listing A.11: Konfiguration der Firewall-Regeln „rules“

```
1  #ACTION         SOURCE  DEST            PROTO   DEST    SOURCE
2  #                                               PORT    PORT(S)
3
4  # Generell
5
6  # Erlaube Pings für alle
7  ACCEPT          all     all             icmp
8
9  # Erlaube DNS-Anfragen aus dem WLAN
10 AllowDNS        wlan    net
11
12 # Abgehend von IXDPG425 (fw - firewall)
13
14 # Network Time Protocol, DNS, Traceroute
15 AllowNTP        fw      net
16 AllowDNS        fw      net
17 AllowTrcrt      fw      net
18
19 # HTTP
20 ACCEPT          fw      net             tcp     80
21
22 # Ankommend an IXDPG425 (fw - firewall)
23
24 # Erlaube HTTP auf Testwebserver
25 ACCEPT          wlan    fw              tcp     80
26 ACCEPT          lan     fw              tcp     80
27 ACCEPT          vpn     fw              tcp     80
```

```
28
29 # Erlaube SSH zur Fernwartung
30 AllowSSH        wlan    fw
31 AllowSSH        lan     fw
32 AllowSSH        vpn     fw
33
34 # Erlaube IPSEC
35 ACCEPT         wlan    fw              tcp     50
36 ACCEPT         wlan    fw              udp     500
37 ACCEPT         all     fw              udp     1701
38 ACCEPT         wlan    fw              50
39
40 # Erlaube DNS-Weiterleitungsanfragen
41 AllowDNS        wlan    fw
42 AllowDNS        vpn     fw
```

Anhang B

Inhalt der beiliegenden DVD

Dieser Diplomarbeit liegt eine DVD bei, auf der die Ergebnisse der praktischen Umsetzung nachvollzogen werden können.

Eine Auswahl der wichtigsten Verzeichnisse zeigt die Tabelle B.1.

Verzeichnis	**Beschreibung**
/rootfs	Wurzeldateisystem des IXDPG425
/tftpboot	Verzeichnis für die per TFTP zu übertragenen Kernel
/slug	Hauptverzeichnis der OpenEmbedded-Entwicklungsumgebung
/slug/bitbake	Das Bitbake-Programm
/slug/ixp	Arbeitsordner für die zu erstellende Linux Distribution
/slug/ixp/build/conf	Speicherort der Hauptkonfigurationsdatei local.conf
/slug/ixp/build/tmp	Ordner zur Ablage sämtlicher im Entwicklungsprozeß entstehender Dateien
/slug/ixp/build/tmp/cache	Zwischenspeicher für Paketbeschreibungen
/slug/ixp/build/tmp/cross	Der Cross Compiler
/slug/ixp/build/tmp/work	Arbeitsverzeichnis für den Cross Compile-Prozeß
/slug/openembedded	Programm- und Konfigurationsdateien sowie Repository für OpenEmbedded
/slug/openembedded/ conf/distro	Speicherort der Distributionsbeschreibung openixp425.conf
/slug/openembedded/ conf/machine	Speicherort der Gerätekonfiguration ixp425.conf
/slug/openembedded/ packages	Ablageort der Paketbeschreibungen (OpenEmbedded Repository)
/slug/source	Quellcode-Archive der eingesetzten Softwaremodule

Tabelle B.1: Inhalt der beiliegenden DVD

Anhang C

Abkürzungsverzeichnis

3DES Triple Data Encryption Standard

AAI Authentication Algorithm Identification

AES Advanced Encryption Standard

AH Authentication Header

AP Access Point

API Application Programming Interface

CA Certificate Authority

CBC Cipher Block Chaining

CCMP CTR/CBC-MAC Protocol

CHAP Challenge Handshake Authentication Protocol

CRC Cyclic Redundancy Check

CTR Counter Mode

DES Data Encryption Standard

DHCP Dynamic Host Configuration Protocol

DNS Domain Name System

DSSS Direct Sequence Spread Spectrum

DSL Digital Subscriber Line

EAP Extended Authentication Protocol

EAPOL EAP over LAN

EAPOW EAP over Wireless

ESP Encapsulation Security Payload

FCC Federal Communications Commission

FCS Frame Check Sequence

FHSS Frequency Hopping Spread Spectrum

FIPS Federal Information Processing Standards

GPL GNU General Public License

HAL Hardware Access Layer

ICV Integrity Check Value

ICMP Internet Control Message Protocol

IDEA International Data Encryption Standard

IEEE Institute of Electrical and Electronics Engineers

IETF Internet Engineering Task Force

IKE Internet Key Exchange

IP Internet Protocol

IPSec Internet Protocol Security

ISAKMP Internet Security Association and Key Management Protocol

ISM-Band Industrial, Scientific and Medical Band

ISO International Organization for Standardization

ITU International Telecommunication Union

IV Initialisierungsvektor

L2TP Layer2 Transport Protokoll

LAN Local Area Network

MAC Media Access Control

MIC Message Integrity Code

NAT Network Address Translation

NIST National Institute of Standards and Technology

NFS Network File System

NPE Network Processor Engine

NTP Network Time Protocol

OCF OpenBSD/FreeBSD Cryptographic Framework

OSI Open Systems Interconnection

PAE Port Access Entity

PAP Password Authentication Protocol

PDA Personal Digital Assistant

PEAP Protected EAP

PKI Public-Key-Infrastruktur

PMK Pairwise Master Key

PPP Point-to-Point Protocol

PPPoE PPP over Ethernet

PSK Pre-Shared Keys

QoS Quality of Service

RADIUS Remote Authentication Dial-In User Service

RC4 Ron's Cipher 4

RFC Request for Comments

RSN Robust Secure Network

SA Security Association

SOHO Small Office / Home Office

SN Sequenznummer

SNMP Simple Network Management Protocol

SSID Service Set Identifier

SSH Secure Shell

TCP Transmission Control Protocol

TFTP Trivial File Transfer Protocol

TKIP Temporal Key Integrity Protocol

TLS Transport Layer Security

TTLS Tunneled Transport Layer Security

UDP User Datagram Protocol

USB Universal Serial Bus

VoIP Voice over IP

VPN Virtual Private Network

WDS Wireless Distribution System

WEP Wired Equivalent Privacy

WiFi Wireless Fidelity

WLAN Wireless LAN

WPA WiFi Protected Access

Anhang D

Literaturverzeichnis

[Bar04] Barth, W. *VPN-Anbindung von Windows-Clients an FreeS/WAN*. April 2004

[BC01] Bovet, D. ; Cesati, M.: *Understanding the Linux Kernel*. Köln : O'Reilly Verlag, 2001

[BK] Burnside, M. ; Keromytis, A. D. *Accelerating Application-Level Security Protocols*

[Eas05] Eastep, T.: *HowTo: IPSEC using Linux Kernel 2.6*. 2005. – http://www.shorewall.net/IPSEC-2.6.html

[Fel04] Feldmann, T.: *Extensible Authentication Protocol*, Universität Koblenz-Landau, Seminararbeit, 2004

[Fie04] Fiegert, J. *HostAP WPA Workshop*. Dezember 2004

[Fre03] Freud, F. *WEP, Sicherheitsanalyse und mögliche Attacken*. 2003

[Fre04] The FreeRADIUS Project: *Homepage The FreeRADIUS Server Project*. 2004. – http://www.freeradius.org/

[Fun04] Funkwerk Enterprise Communications GmbH: *Benutzerhandbuch Bintec X2205*. April 2004

[Fun05] Funkwerk Enterprise Communications GmbH: *Produktbeschreibung Bintec X2250*. 2005. – http://www.funkwerk-ec.com/prod_bintec_x2250_main_de,15404,194.html

[Hag03] Hagedorn, A.: *IEEE 802.11i - Sicherheit in drahtlosen lokalen Netzen*, TU Darmstadt, Diplomarbeit, November 2003

[Int04a] Intel Corporation: *Product Brief „Intel® IXP425 Network Processor“*. 2004

[Int04b] Intel Corporation: *User's Guide „Intel® IXDPG425 Gateway Reference Platform“*. November 2004

[Ive03] Ivers, M.: *Netzwerkauthentizierung mit IEEE 802.1X*, Technische Universität Braunschweig, Seminararbeit, 2003

[Jun05] Jungo Ltd.: *Produktbeschreibung OpenRG*. 2005. – http://www.jungo.com/openrg/products.html

[Kar04] Kartes, C.: Drahtlose Netze optimal absichern. In: *Funkschau* 22/2004 (2004)

[KWR03] Keromytis, A. D. ; Wright, J. L. ; de Raadt, T. *The Design of the OpenBSD Cryptographic Framework*. Juni 2003

[Lac02] Lachmund, S.: *IPSec - Sicherheitsaspekte und Konfiguration unter Windows 2000 und Linux*, Fachhochschule München, Diplomarbeit, September 2002

[LAN04] LANCOM. *LANCOM Techpaper - WPA und 802.11i*. 2004

[Lau05] Lauer, M. *Building Embedded Linux Distributions with Bake and OpenEmbedded*. Februar 2005

[Lef03] Leffler, S. J. *Cryptographic Device Support for FreeBSD*. 2003

[Mal05a] Malinen, J.: *Host AP driver for Intersil Prism2/2.5/3, hostapd, and WPA Supplicant*. 2005. – http://hostap.epitest.fi/

[Mal05b] Malinen, J.: *Linux WPA/WPA2/IEEE 802.1X Supplicant*. 2005. – http://hostap.epitest.fi/wpa_supplicant/

[McC05] McCullough, D.: *Homepage OCF-Projekt*. 2005. – http://ocf-linux.sourceforge.net/

[Mül03] Müller, M.: *Homepage VPN Tool*. 2003. – http://vpn.ebootis.de/

[Mon05] MontaVista Software: *Produktbeschreibung MontaVista Linux Professional Edition*. 2005. – http://www.mvista.com/products/pro/features.html

[NMG01] Nett, E. ; Mock, M. ; Gergeleit, M.: *Das drahtlose Ethernet*. München : Addison-Wesley Verlag, 2001

[Orm98] Orman, H.: *Request for Comments: 2412*. 1998. – http://www.faqs.org/rfcs/rfc2412.html

[Pro04a] Project OpenCA: *Homepage The OpenCA Project*. 2004. – http://www.openca.info

[Pro04b] Project RP-L2TP: *Homepage RP-L2TP Projekt*. 2004. – http://sourceforge.net/projects/rp-l2tp/

[Rec04] Rech, J.: *802.11-WLAN-Technologie und praktische Umsetung im Detail ; 802.11a/h, 802.11b, 802.11g, 802.11i, 802.11e*. Hannover : Heise Verlag, 2004

[Ros01] Roser, K. *HOWTO: EAP/TLS Setup for FreeRADIUS and Windows XP Supplicant*. April 2001

[Sch05] Schmitz, U.: Weltpolitik beeinflusst die Internet-Sicherheit. In: *Computer Zeitung* 27/2005 (2005)

[Ste04] Steffen, A.: Sicherheit von Wireless LAN: Attacken und Schutzmassnahmen. In: *SEV Bulletin / SEV Verband für Elektro-, Energie- und Informationstechnik (Hrsg.)* 11 (2004)

[Ste05] Stepping, C.: *Drahtlose Netze*. Wilburgstetten : Schlembach Verlag, 2005

[TY03] Tseng, T. H. ; Ye, F.: Linux Access Point and IPSec Bridge. In: *Tamkang Journal of Science and Engineering* 6/2003 (2003), S. 121–126

[Win04] Winter, S.: *Multihop-Zugang in IEEE 802.11 basierten Funknetzen*, Universität Karlsruhe, Seminararbeit, September 2004

Selbstständigkeitserklärung

Hiermit versichere ich, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie Zitate kenntlich gemacht habe.

Daniel Woithe

Dresden, 11. August 2005